



**Defense Special Weapons Agency  
Alexandria, VA 22310-3398**



**DSWA-TR-97-15**

## **Tester Operating Manual and Software Description**

**H. Jake Tausch  
David Sleeter  
Mission Research Corp.  
1720 Randolph Road, SE  
Albuquerque, NM 87106-4245**

**October 1997**

*EXCLUDED FROM AUTOMATIC DOWNGRADING AND DECLASSIFICATION*

**Technical Report**

**CONTRACT No. DNA 001-92-C-0152**

**Approved for public release;  
distribution is unlimited.**

**19971028 020**

**DESTRUCTION NOTICE:**

Destroy this report when it is no longer needed.  
Do not return to sender.

PLEASE NOTIFY THE DEFENSE SPECIAL WEAPONS  
AGENCY, ATTN: CSTI, 6801 TELEGRAPH ROAD,  
ALEXANDRIA, VA 22310-3398, IF YOUR ADDRESS IS  
INCORRECT, IF YOU WISH IT DELETED FROM THE  
DISTRIBUTION LIST, OR IF THE ADDRESSEE IS NO  
LONGER EMPLOYED BY YOUR ORGANIZATION.



## DISTRIBUTION LIST UPDATE

This mailer is provided to enable DSWA to maintain current distribution lists for reports. (We would appreciate your providing the requested information.)

- ☐ Add the individual listed to your distribution list.
- ☐ Delete the cited organization/individual.
- ☐ Change of address.

### NOTE:

Please return the mailing label from the document so that any additions, changes, corrections or deletions can be made easily. For distribution cancellation or more information call DSWA/IMAS (703) 325-1036.

NAME: \_\_\_\_\_

ORGANIZATION: \_\_\_\_\_

### OLD ADDRESS

### CURRENT ADDRESS

---

---

---

---

---

---

TELEPHONE NUMBER: (     ) \_\_\_\_\_

### DSWA PUBLICATION NUMBER/TITLE

### CHANGES/DELETIONS/ADDITIONS, etc.)

(Attach Sheet if more Space is Required)

---

---

---

---

---

---

DSWA OR OTHER GOVERNMENT CONTRACT NUMBER: \_\_\_\_\_

CERTIFICATION OF NEED-TO-KNOW BY GOVERNMENT SPONSOR (if other than DSWA):

SPONSORING ORGANIZATION: \_\_\_\_\_

CONTRACTING OFFICER OR REPRESENTATIVE: \_\_\_\_\_

SIGNATURE: \_\_\_\_\_

CUT HERE AND RETURN



DEFENSE SPECIAL WEAPONS AGENCY  
ATTN: IMAS  
6801 TELEGRAPH ROAD  
ALEXANDRIA, VA 22310-3398

DEFENSE SPECIAL WEAPONS AGENCY  
ATTN: IMAS  
6801 TELEGRAPH ROAD  
ALEXANDRIA, VA 22310-3398

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE 971001		3. REPORT TYPE AND DATES COVERED Technical 920904 - 960729
4. TITLE AND SUBTITLE Tester Operating Manual and Software Description			5. FUNDING NUMBERS C - DNA 001-92-C-0152 PE - 62715H PR - AF TA - DE WU - DH00030	
6. AUTHOR(S) H. Jake Tausch and David Sleeter				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Mission Research Corp. 1720 Randolph Road, SE Albuquerque, NM 878106-4245			8. PERFORMING ORGANIZATION REPORT NUMBER  MRC/ABQ-R-1809	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Special Weapons Agency 6801 Telegraph Road Alexandria, VA 22310-3398 ESE/Cohn			10. SPONSORING/MONITORING AGENCY REPORT NUMBER  DSWA-TR-97-15	
11. SUPPLEMENTARY NOTES This work was sponsored by the Defense Special Weapons Agency under RDT&E RMC Code B4662D AF DE 00030 7010A 25904D.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT  Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words)  This program developed a high speed portable integrated circuit tester which can generate sophisticated memory patterns and is capable of performing <i>in-situ</i> diagnostics to facilitate the identification of circuit failures. This document consists of the tester operating manual and programmer's guide..				
14. SUBJECT TERMS ASIC Testing Integrated Circuit  Memory Radiation Testing			15. NUMBER OF PAGES 46	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT  SAR	

**UNCLASSIFIED**

**SECURITY CLASSIFICATION OF THIS PAGE**

**CLASSIFIED BY:**

N/A since Unclassified.

**DECLASSIFY ON:**

N/A since Unclassified.

**CLASSIFICATION OF THIS PAGE**  
**UNCLASSIFIED**

## SUMMARY

The purpose of this program was to develop and demonstrate a portable integrated circuit tester to support radiation testing (e.g. SEU, dose-rate upset, and survivability, total dose testing, etc.). Radiation testing is an integral part of the development of high performance radiation hardened microelectronics. Prior to this program there existed no portable integrated circuit tester which provided the required excitation and was capable of performing in-situ diagnostics at radiation facilities at functional speeds designed for the test devices.

This program developed a high speed integrated circuit memory tester which is readily portable, can generate sophisticated memory patterns, and is capable of performing in-situ diagnostics to facilitate the identification of circuit failures. This tester is called the Algorithmic/Test Vector<sup>TM</sup> (ATV<sup>TM</sup>) system.

The compact size and high performance of the tester was largely achieved through the use of a full custom ASIC (Application Specific Integrated Circuit). This ASIC provides 8 channels of high speed formatting, delay and co-axial drive and replaces the equivalent of 24 integrated circuits.

The design has a simple, user friendly graphical interface which allows quick and easy programming of the system.

Effectiveness of the tester was demonstrated in May of 1996 when it was used to support heavy ion testing at the Brookhaven National Laboratory for the Radiation Tolerant Microelectronics program (DNA001-95-C-0041). In this program National Semiconductor Corporation produced CLAY31 configurable logic arrays which then needed to be characterized for SEU. Testing of the CLAY31s involved downloading a very complex programming file to the parts, periodically checking them for upsets, then re-programming the parts and logging errors when upsets were detected. The test was completely successful and demonstrated the effectiveness of the ATV tester.

ATV systems will provide advanced testing and diagnostic capabilities to users in the radiation effects community and will also be a valuable alternative to commercial firms that produce or test memories, ASICs, CPLDs, or any logic device. We have developed a detailed plan for marketing and selling these systems. Key aspects of the plan include minimizing maintenance support requirements by simplifying user interactions, extensive on-line help capability, and repair of systems through board replacement.

# CONVERSION TABLE

Conversion factors for U.S. Customary to metric (SI) units of measurement.

MULTIPLY \_\_\_\_\_ BY \_\_\_\_\_ TO GET  
TO GET < \_\_\_\_\_ BY < \_\_\_\_\_ DIVIDE

angstrom	1.000 000 × E -10	meters (m)
atmosphere (normal)	1.013 25 × E +2	kilopascal (kPa)
bar	1.000 000 × E +2	kilopascal (kPa)
barn	1.000 000 × E -28	meter <sup>2</sup> (m <sup>2</sup> )
British thermal unit (thermochemical)	1.054 350 × E +3	joule (J)
calorie (thermochemical)	4.184 000	joule (J)
cal (thermochemical/cm <sup>2</sup> )	4.184 000 × E -2	megajoule/m <sup>2</sup> (MJ/m <sup>2</sup> )
curie	3.700 000 × E +1	*gigabecquerel (GBq)
degree (angle)	1.745 329 × E -2	radian (rad)
degree Fahrenheit	$t_k = (t_f + 459.67)/1.8$	kelvin (K)
electron volt	1.602 19 × E -19	joule (J)
erg	1.000 000 × E -7	joule (J)
erg/second	1.000 000 × E -7	watt (W)
foot	3.048 000 × E -1	meter (m)
foot-pound-force	1.355 818	joule (J)
gallon (U.S. liquid)	3.785 412 × E -3	meter <sup>3</sup> (m <sup>3</sup> )
inch	2.540 000 × E -2	meter (m)
jerk	1.000 000 × E +9	joule (J)
joule/kilogram (J/kg) radiation dose absorbed	1.000 000	Gray (Gy)
kilotons	4.183	terajoules
kip (1000 lbf)	4.448 222 × E +3	newton (N)
kip/inch <sup>2</sup> (ksi)	6.894 757 × E +3	kilopascal (kPa)
ktop	1.000 000 × E +2	newton-second/m <sup>2</sup> (N·s/m <sup>2</sup> )
micron	1.000 000 × E -6	meter (m)
mil	2.540 000 × E -5	meter (m)
mile (international)	1.609 344 × E +3	meter (m)
ounce	2.834 952 × E -2	kilogram (kg)
pound-force (lbs avoirdupois)	4.448 222	newton (N)
pound-force inch	1.129 848 × E -1	newton-meter (N·m)
pound-force/inch	1.751 268 × E +2	newton/meter (N/m)
pound-force/foot <sup>2</sup>	4.788 026 × E -2	kilopascal (kPa)
pound-force/inch <sup>2</sup> (psi)	6.894 757	kilopascal (kPa)
pound-mass (lbm avoirdupois)	4.535 924 × E -1	kilogram (kg)
pound-mass-foot <sup>2</sup> (moment of inertia)	4.214 011 × E -2	kilogram-meter <sup>2</sup> (kg·m <sup>2</sup> )
pound-mass/foot <sup>3</sup>	1.601 846 × E +1	kilogram/meter <sup>3</sup> (kg/m <sup>3</sup> )
rad (radiation dose absorbed)	1.000 000 × E -2	**Gray (Gy)
roentgen	2.579 760 × E -4	coulomb/kilogram (C/kg)
shake	1.000 000 × E -8	second (s)
slug	1.459 390 × E +1	kilogram (kg)
torr (mm Hg, 0° C)	1.333 22 × E -1	kilopascal (kPa)

\*The becquerel (Bq) is the SI unit of radioactivity; 1 Bq = 1 event/s.

\*\*The Gray (Gy) is the SI unit of absorbed radiation.



## TABLE OF CONTENTS

Section	Page
SUMMARY.....	iii
CONVERSION TABLE.....	iv
FIGURES.....	vi
1 TESTER OPERATING MANUAL.....	1
1.1 HARDWARE DESCRIPTION .....	1
1.1.1 Adjustments.....	5
1.1.2 Expanding to Multiple Chassis.....	7
1.2 PROGRAMMABLE THEVININ LOAD.....	7
1.3 HOW TO RUN A TEST .....	9
1.3.1 Test Fixtures.....	10
1.3.2 Part Pin Assignment .....	17
1.3.3 Making Timing Diagrams .....	19
1.3.4 Algorithm Project.....	22
1.3.5 Creating an Algorithm.....	24
1.3.6 Test Execution.....	26
1.3.7 Data Log .....	28
2 PROGRAMMERS GUIDE .....	30

## FIGURES

Figure		Page
1-1	System interconnections.....	2
1-2	Functional block diagram of the TVP board.....	3
1-3	Functional block diagram of the data board.....	4
1-4	Functional block diagram of the address board.....	4
1-5	Physical overview of TVP board.....	5
1-6	Physical overview of data board.....	6
1-7	Physical overview of address board.....	7
1-8	Thevinin load.....	8
1-9	Thevinin termination circuit.....	9
1-10	ATV combines test elements based on pin/pin group names.....	10
1-11	Sample test fixture.....	11
1-12	Excerpt from a test fixture ASCII file.....	12
1-13	Test fixture and pin assignment.....	17
1-14	Timing diagram dialog box.....	20
1-15	Algorithm project dialog box.....	23
1-16	Text editor with ATV algorithm template code.....	25
1-17	Sample algorithm code.....	25
1-18	Data log setup.....	27
1-19	Run algorithm dialog box.....	28
1-20	Sample data log.....	29
2-1	Syntax chart of program flow.....	31
2-2	Syntax chart of a program block.....	32
2-3	Syntax chart for high level instructions.....	33
2-4	Syntax chart for assembly level instructions.....	34

## SECTION 1 TESTER OPERATING MANUAL

### 1.1 HARDWARE DESCRIPTION.

An ATV system consists of a PC type controlling computer and one or more chassis containing ATV cards. There are four types of ATV cards: Test Vector Processor (TVP) Card, Hub Card, 16 channel I/O card, and 24 channel Output Only Card. Each system requires one TVP card and each chassis requires one Hub Card. Other chassis slots can be used for I/O and Output Only cards.

The ATV system was designed to achieve 50 MHz algorithmic flow with all system output signals synchronized to within several hundred pico-seconds, even when system cards are distributed across several chassis. This is achieved by separately controlling program flow and high speed timing.

Program flow is controlled by the Test Vector Processor (TVP) card which generates algorithm control information for other cards in the system. This information is distributed throughout the system via the hub cards. In the main chassis the TVP card is connected to the hub card and sends the algorithm control information to it. From there the data is distributed to hub cards in other chassis and from each hub card to the ATV cards in that particular chassis.

High speed timing is controlled within each chassis by a high speed, multi-phase clock located on each hub card. Clocks on the hub cards are phase-locked together so that timing accuracy is maintained across all chassis. Algorithmic information is combined with the high speed clocks in custom ASICs to generate the high speed output signals. These Format and Delay ASICs (FNDs) generate the formatted signals (e.g. Return To Complement), and provide individually adjustable delays for every pin/waveform.

Figure 1-1 is an overview of a typical system which shows how the various cards interact. Note in this figure that there is a controlling PC and two ATV chassis. The PC communicates with the ATV cards via expansion cards, one of which is located in the PC and one in each of the chassis. The PC uses this link to program ATV cards, to upload error patterns after a test, and to handshake with the TVP during a test.

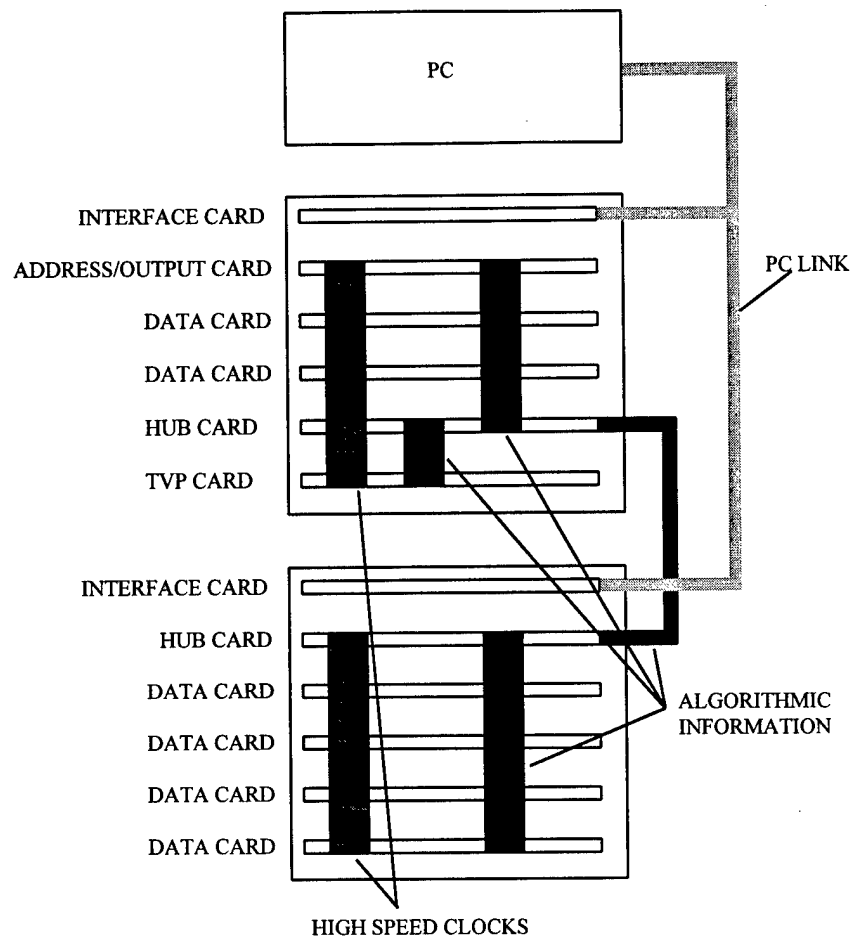


Figure 1-1. System interconnections.

Figure 1-2 is a functional block diagram of the TVP board. The main function of the TVP board is to control the algorithmic test flow of the system. The Test Vector Processor itself is a custom RISC processor that executes programs from on-board memory. The system PC downloads this memory and starts the TVP running at the beginning of a test. Once it is running, the TVP and PC can communicate with each other through an 8 bit mailbox to co-ordinate such things as start-up conditions, end of test, change bias, etc.

The TVP card has two separately programmable voltage sources with programmable current limits that can be used to bias up test devices. One of the voltage sources also has an IDDQ detection circuit.

The TVP card also has TTL level trigger-out and trigger-in signals that can control external circuitry or be used to co-ordinate algorithmic execution based on external events.

The TVP card also has 6 general purpose output strobes that can be used

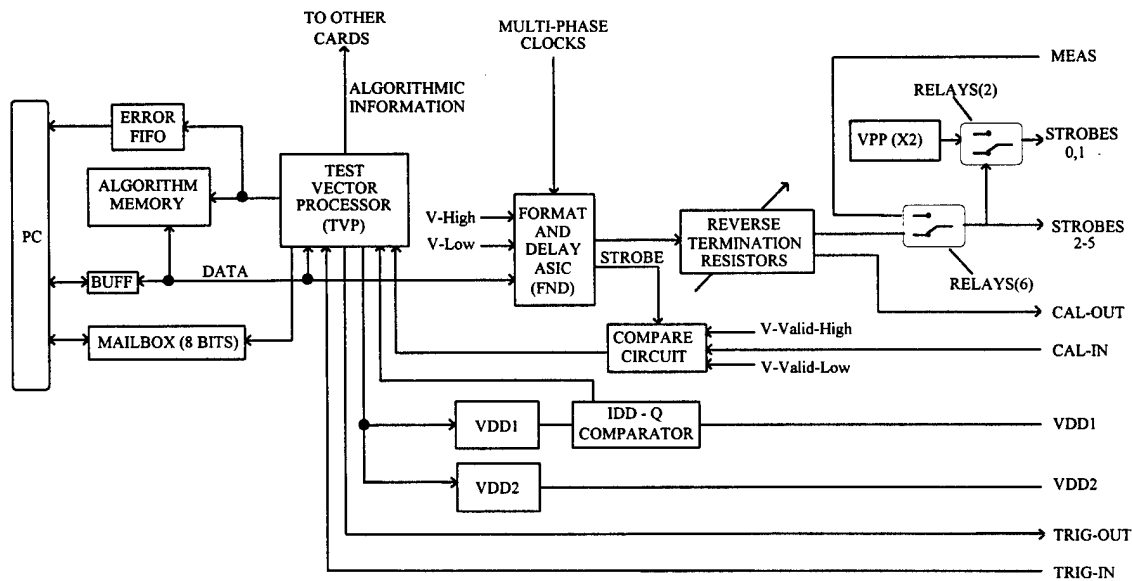


Figure 1-2. Functional block diagram of the TVP board.

Two of the strobe outputs can be connected through relays to programmable VPP sources for programming devices.

For separate stimulation or analog characterization, any combination of the strobe output pins can be connected through relays to an SMB connector (MEAS) on the rear of the card.

Two signals are provided over SMB connectors for high speed calibration. CAL-OUT is a 50 ohm reverse terminated strobe that provides stimulation and CAL-IN is a 90 ohm terminated input to a high speed comparator circuit.

The functional diagram of a data board is shown in Figure 1-3. Algorithmic information is used by this board to generate output data patterns and to compare data patterns against data received from the DUT. When an error is detected, the patterns of expected data and errors are stored in a FIFO and later read by the PC.

For separate stimulation or analog characterization, any combination of the data output pins can be connected through relays to an SMB connector (MEAS) on the rear of the card.

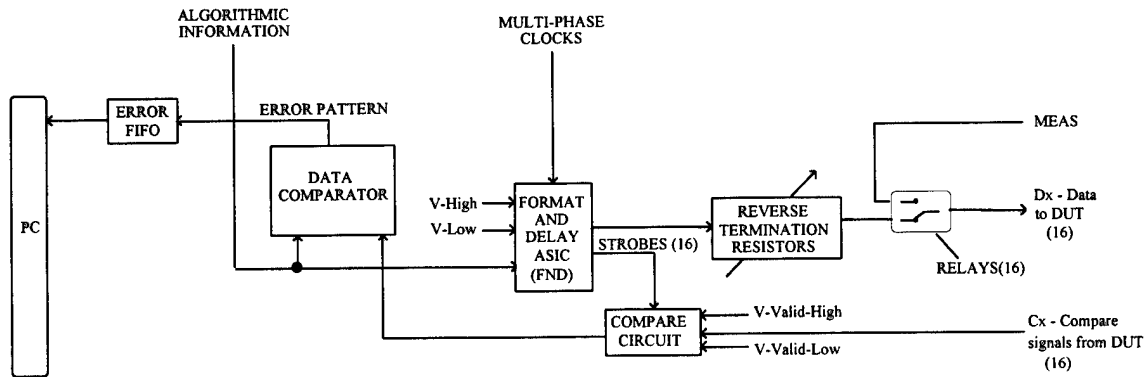


Figure 1-3. Functional block diagram of the data board.

Figure 1-4 is a functional diagram of an address/output only board. Algorithmic information is received by this card and used to generate data patterns output to the DUT. The address processor on this card can output patterns directly, or can output patterns generated by an internal up/down counter. The address processor also has an internal arithmetic logic unit which can perform mathematical operations on the pattern stored in the counter. The address processor also has the ability to multiplex high and low counter values onto the lower order pins for multiplexing RAS/CAS values in DRAMs.

For separate stimulation or analog characterization, any combination of the data output pins can be connected through relays to an SMB connector (MEAS) on the rear of the card.

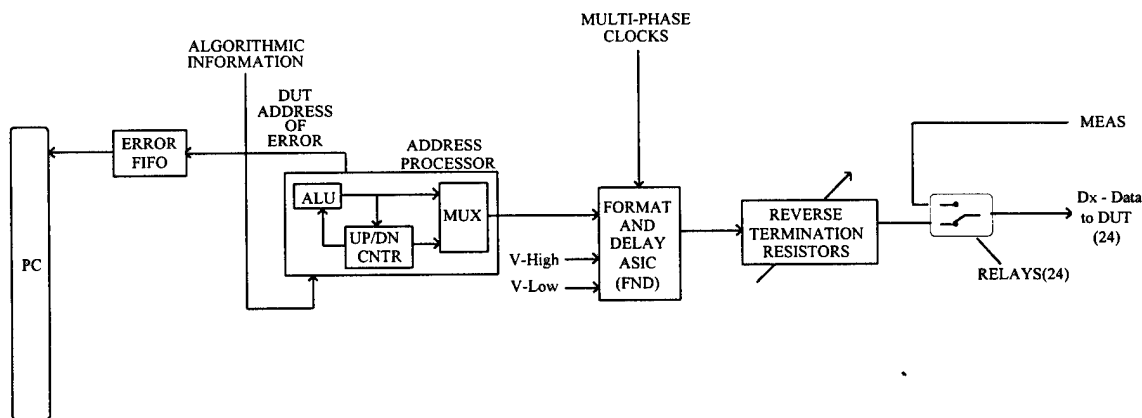


Figure 1-4. Functional block diagram of the address board.

### 1.1.1 Adjustments.

There are three types of adjustments required for each card. The card's address must be set using dip switch selectors, each card's reference voltage must be adjusted, and the reverse terminate impedance must be trimmed on each high-speed output pin.

Reference voltages are adjusted by connecting a DVM to the appropriate test point on each card and adjusting a trim pot to set the measured voltage to 9 volts.

Reverse terminate impedance can best be adjusted by observing output signals at the end of a coaxial cable and adjusting the appropriate trim pot.

The following figures provide details for adjusting each card type.

Figure 1-5 shows the physical location of calibration components on the TVP card and also shows the pinout of signals on the card's rear connector.

The card's reference voltage is adjusted by monitoring the voltage on TP12 while adjusting R109. This voltage should be set to 9 volts.

The reverse termination of each high speed output can be set by adjusting the trim resistor paired with each output on the figure. For instance, channel D0 can be adjusted using R107. On the figure the resistor designator is followed by the letter "b" indicating the resistor is physically located on the opposite side of the board from the one shown.

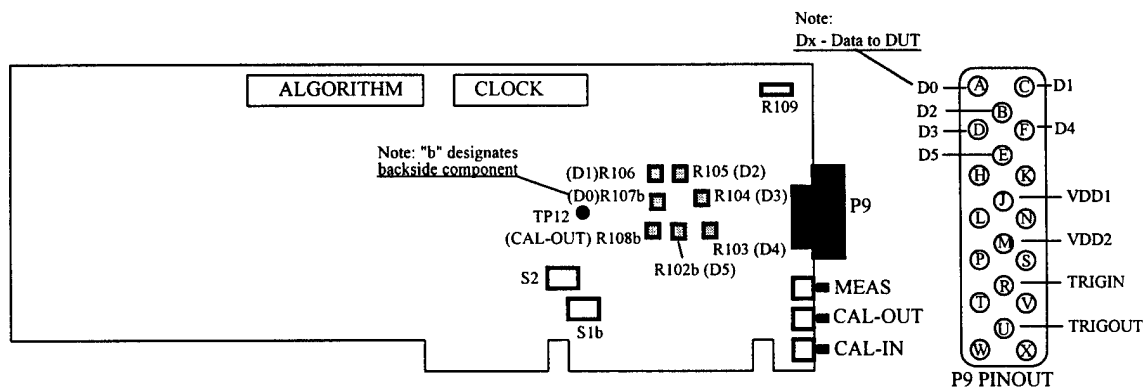


Figure 1-5. Physical overview of TVP board.

Figure 1-6 shows the physical location of calibration components on each data card and also shows the pinout of signals on the card's rear connector.

The card's reference voltage is adjusted by monitoring the voltage on TP11 while adjusting R15. This voltage should be set to 9 volts.

The reverse termination of each high speed output can be set by adjusting the trim resistor paired with each output on the figure. For instance, channel D0 can be adjusted using R31. Resistor designators on the figure that are followed by the letter "b" indicate the resistor is physically located on the opposite side of the board from the one shown.

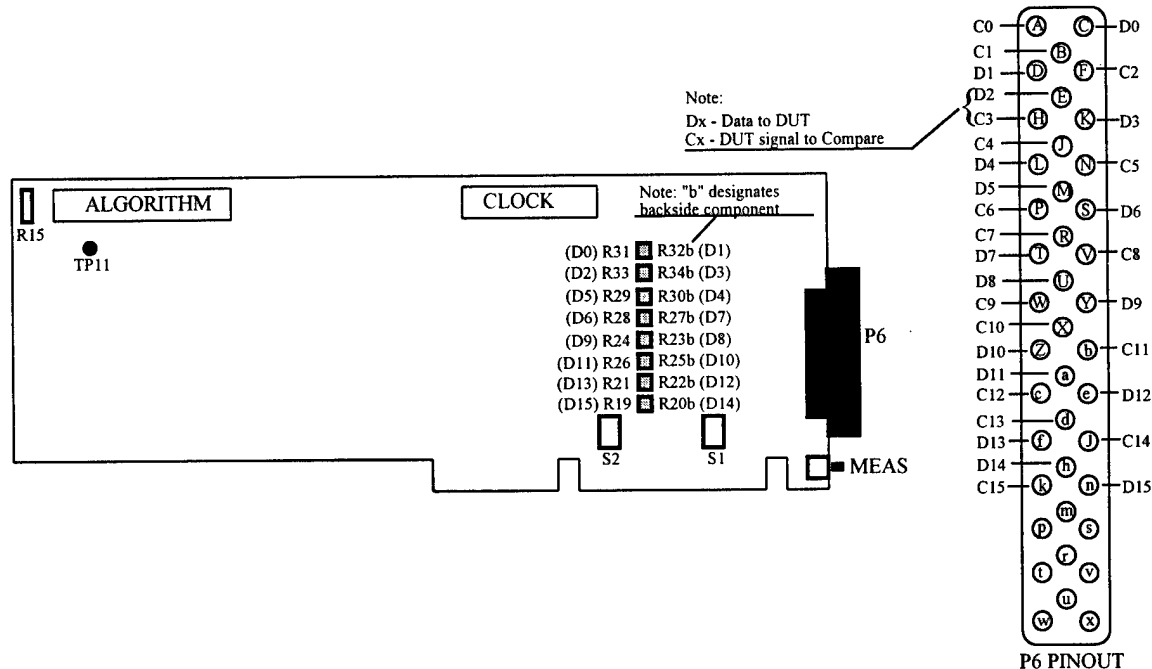


Figure 1-6. Physical overview of data board.

Figure 1-7 shows the physical location of calibration components on each data card and also shows the pinout of signals on the cards rear connector.

The card's reference voltage is adjusted by monitoring the voltage on TP9 while adjusting R37. This voltage should be set to 9 volts.

The reverse termination of each high speed output can be set by adjusting the trim resistor paired with each output on the figure. For instance, channel D0 can be adjusted using R53. Resistor designators on the figure that are followed by the letter "b" indicate the resistor is physically located on the opposite side of the board from the one shown.



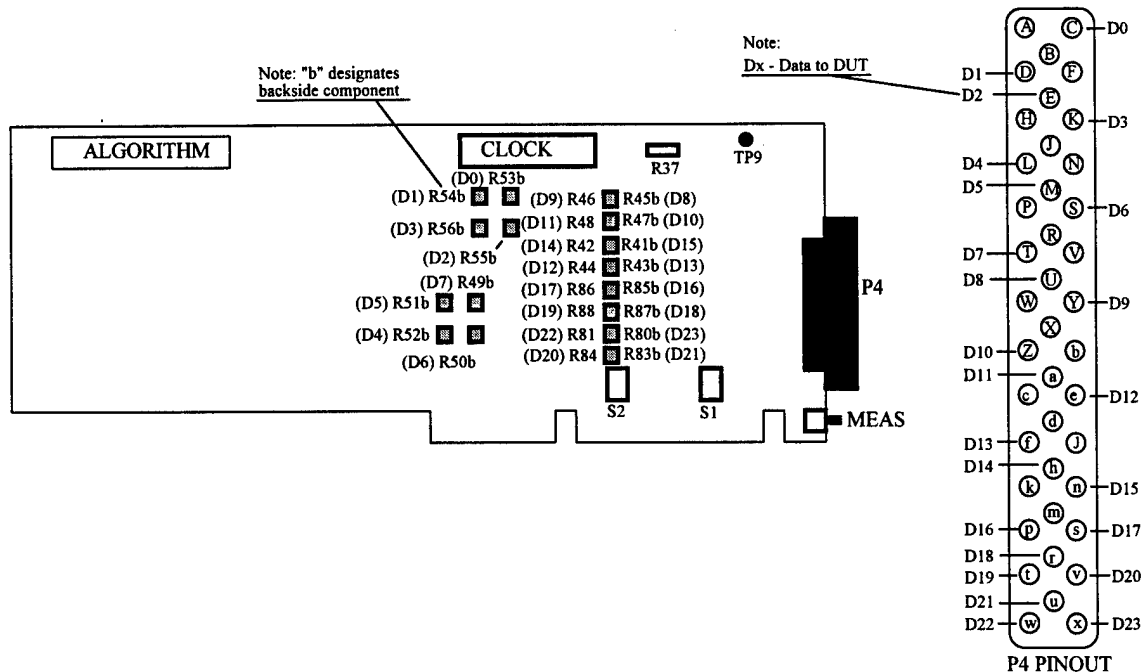


Figure 1-7. Physical overview of address board.

### 1.1.2 Expanding to Multiple Chassis.

In multiple chassis configurations the individual Hub Cards in each chassis will be inter-connected with and phase-locked to each other so that timing will be maintained for all output pins. One Hub Card is designated the master by installing jumpers J3, J4, J5, J6 and J7. Other Hub cards will be configured as slaves by removing these jumpers. Interconnections are made by connecting a co-axial cable between the SMB connectors labeled P4 on the rear of the hub cards and by connecting a 60 pin ribbon cable between the ribbon connectors labeled P5 on the rear of the hub cards.

## 1.2 PROGRAMMABLE THEVININ LOAD.

A Thevinin load consists of a series resistance back terminated in a voltage source as shown in Figure 1-8. Given any two current/voltage bias conditions, a Thevinin load can be calculated which will produce those conditions.

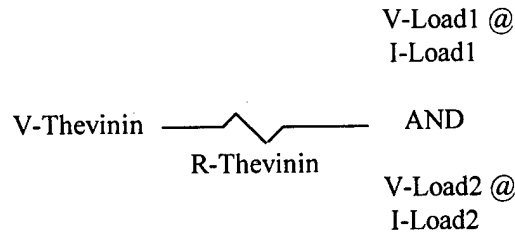


Figure 1-8. Thevinin load.

The Thevinin resistance and voltage for a given load can be calculated as follows.

$$I\text{-load1} * R\text{-Thevinin} = (V\text{-load1} - V\text{-Thevinin})$$

$$I\text{-load2} * R\text{-Thevinin} = (V\text{-load2} - V\text{-Thevinin})$$

Subtracting:

$$R\text{-Thevinin} = (V\text{-load1} - V\text{-load2}) / (I\text{-load1} - I\text{-load2})$$

And:

$$V\text{-Thevinin} = V\text{-load1} - I\text{-load1} * R\text{-Thevinin}$$

A typical part might want to have a load of +5mA when its output was at 4.5V and a load of -15mA when its output was 0.5V. From the above equations it is easy to see that R-Thevinin should be 200 ohms and V-Thevinin should be 1.5V.

Each ATV I/O channel has two co-axial connections. A back terminated 50 ohm driver outputs voltages to provide logic high and logic low signals to the DUT and also to provide a third voltage for the Thevinin load when the DUT is outputting a signal. Signals are returned from the DUT over a separate 90 ohm coax cable which is resistively terminated. These co-axial signals are typically tied together at the DUT card as shown in Figure 1-9. This figure also shows how a resistor can be added to the DUT card to complete the Thevinin load.

The resistive load as seen by the DUT will be the resistor on the DUT card plus the parallel combination of 50 and 90 ohms (32 ohms). In the above example where R-Thevinin was 200 ohms, a resistor of 168 ohms would be placed on the DUT card in series with each output. The equivalent Thevinin voltage as seen by the DUT will be the voltage from the 50 ohm driver times 0.643 which is the voltage divider ratio of the 50 and 90 ohm circuits. In the above example the 50 ohm drive voltage would be 2.33V to produce a Thevinin voltage of 1.5V.

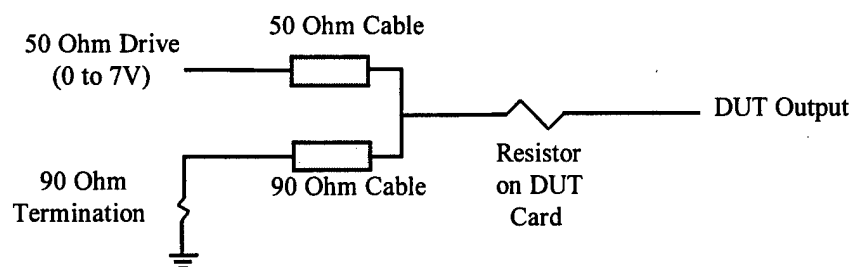


Figure 1-9. Thevinin termination circuit.

Note that placing a 168 ohm resistor in series with each DUT I/O pin should not have any significant effect when sending data to the DUT because the inputs of modern circuits typically have very high impedances. The resistor will affect the magnitude of the DUT signal returned to the I/O channel and this must be taken into account. From simple circuit analysis it can be seen that the voltage at the 90 ohm termination (input to the voltage comparators) will be:

$$V\text{-Comparator} = V\text{-Thevinin} + (32 / (32 + R\text{-Dut})) * (V\text{-Dut} - V\text{-Thevinin})$$

Again using the above example, assume the specification for valid high DUT signals was 4V and for valid low signals was 1V. The actual comparator levels would then be set to 1.9V and 1.42V respectively. Thus, the actual voltage difference between valid high and low at the DUT was 3V whereas the programmed difference at the comparators was 0.48V.

### 1.3 HOW TO RUN A TEST.

An ATV test consists of three independently generated components: a test fixture with pin assignments, timing diagrams and an algorithmic program. Information from these components are combined based on Pin and PinGroup names as illustrated below in Figure 1-10. In this example a group of pins in a test fixture are named "DATA." A timing diagram named "WR" has a waveform named "DATA." A "TDG" line in the algorithm applies the bit pattern "110" to the pins in the "DATA" group using timing from the "DATA" waveform in the "WR" timing diagram.

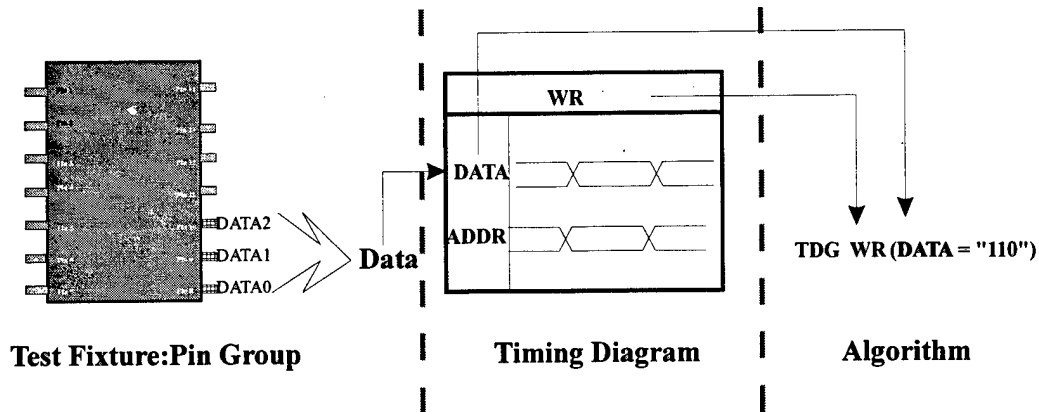


Figure 1-10. ATV combines test elements based on pin/pin group names.

Generally it is a good idea to plan naming conventions before creating any of these components. With proper planning you can combine sets of these components to create many different tests. The details for how to build test fixtures, timing diagrams and algorithms are given below. Once these components are generated a test is run as follows:

A test is created by linking together an algorithm with a text fixture and some number of timing diagrams in an Algorithm Project user interface. Algorithm Projects are accessed by selecting **Instruments|ATV|Algorithm**.

Before running a test, the algorithm must be compiled, and a down load file built and downloaded to ATV. This is accomplished by selecting **Algorithm|Compile**, followed by selecting **Algorithm||Build Download**. Next download the test by selecting **Algorithm|Load**.

The test is then run by selecting **Algorithm|Run** at which point the Run Algorithm Dialog Box appears. Set the test frequency then run the test by pressing the Run button.

Test results are stored in a EXCEL style spread. Prior to running the test, Data Log parameters can be setup by selecting **Algorithm|DataLog Setup**. (Note: The spread sheet is initially displayed as an icon in the bottom left corner of the screen. To view test results while a test is running, double click on the spread sheet icon.)

### 1.3.1 Test Fixtures.

A Test Fixture is used to specify pin names for a Device Under Test (DUT). These pin names are used by loaded instrument drivers to stimulate the proper pins on the DUT. Test fixtures are designed by the user and added to a data base. The test fixture is a graphical element that visually portrays the actual part package, test board or interconnect of the DUT. The following

sections describe how to create a test fixture, and how to make pin name assignments for a specific part in existing test fixtures.

**1.3.1.1 Creating A Test Fixture.** Test fixtures are derived from **ASCII files** with a specific format. The ASCII files can be created externally and imported. Imported ASCII files can be created in a word processor, but are more typically created from a spread sheet. Spread sheet creation is useful and efficient when the placement of pins can be specified as spread sheet formulas. This method may be preferable when there are a large quantity of regularly spaced pins, or when several variations of a test fixture type are to be created. Example test fixture spread sheets are included in the `\examples` subdirectory.

Test Fixtures are composed of basic graphical objects including lines, filled and empty rectangles and ellipses, and text. These objects are used to build a visual image of a test fixture according to dimensions and aspects specified in the formatted ASCII file. The test fixture visually portrays the actual part package, test board or interconnect of the DUT. A sample test fixture is shown in Figure 1-11.

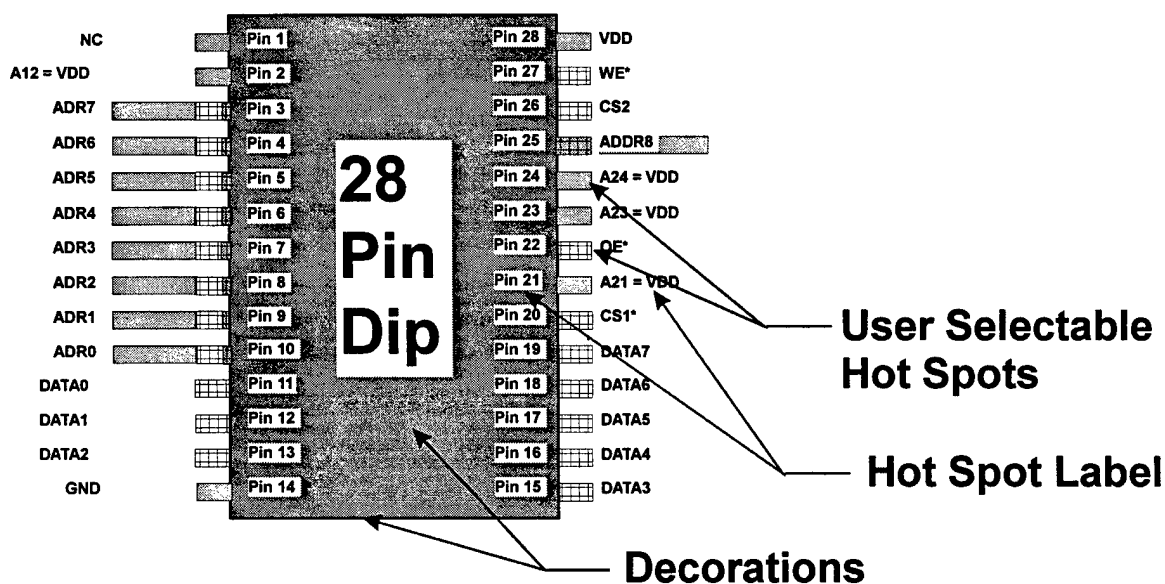


Figure 1-11. Sample test fixture.

Graphic objects in the ASCII file can be one of two types: **Decorations** or **Hot Spots**.

Decorations are inactive objects which the designer can use to help convey the visual look or titles of the test fixture. The picture is also built with definable '*hot spots*' that the user identifies as pins, and are selectable using the mouse. Hot spots are rectangle or ellipse objects that have a **Selectable** flag set to TRUE in the ASCII file. When the user selects a hot spot, it is highlighted along with two associated text objects. These text objects are pin name identifiers which the text

fixture designer uses to cue the user to the function of the selectable pin. One of the text objects is replaced by the pin name that the user will specify when making pin assignments.

Section 1.3.1.2 describes details of the Test Fixture ASCII file format. This provides all the information necessary to construct test fixture files externally. Once a test fixture has been created it can be tested before loading into a data base. See Section 1.3.1.4, Testing And Loading A Text Fixture (\*.fxt) File.

1.3.1.2 Test Fixture ASCII File Format. A sample excerpt from a test fixture ASCII file is given below in Figure 1-12:

```

ATV-TEST-FIXTURE-DRAWING
14 Pin DIP
BEGIN-BLOCK    // Decoration Definitions
0
2
2 0 0 0 0 2 0 0 12632256 0 400 100 700 340
4 0 0 0 0 16777215 550 130 730 355 0 -900 10 14 Pin DIP
END-BLOCK      // End Definition of Decorations
BEGIN-BLOCK    // Pin Definitions for Connector 1
1
I/O 1
INPUT/OUTPUT
32
42
2 1 1 3 0 1 0 0 15780008 5 368 116 400 132
4 1 0 0 0 16777215 415 116 465 132 0 0 5 Pin 1
4 1 0 0 0 16711680 292 116 392 132 2 0 8 PIN (1)
2 2 1 3 0 1 0 0 15780008 5 368 148 400 164
4 2 0 0 0 16777215 415 148 465 164 0 0 5 Pin 2
4 2 0 0 0 16711680 292 148 392 164 2 0 8 PIN (2)

. . . .
2 14 1 3 0 1 0 0 15780008 5 700 308 732 324
4 14 0 0 0 16777215 635 308 685 324 2 0 6 Pin 14
4 14 0 0 0 16711680 740 308 840 324 0 0 9 PIN (14)
END-BLOCK      // End Definition of Pins for Connector 1

```

Figure 1-12. Excerpt from a test fixture ASCII file.

The ASCII file must have a '.fxt' extension.

**Header** - The string 'ATV-TEST-FIXTURE-DRAWING' must appear at the top of the file followed by the test fixture name.

**Blank Lines** - Import error checking gives error messages with line numbers. Blank lines are not counted. If ASCII file includes blank lines, reported error line will not correlate to the file.

**Comments** must be preceded by two forward slashes '//'.

**Blocks** - Graphical objects in the ASCII file are arranged in groups or blocks. Blocks are initiated with the string '**BEGIN-BLOCK**' and end with the string '**END-BLOCK**'. There are two types of blocks defined as follows:

<i>block type</i>	<i>ascii code</i>
Decoration	0
Connector	1

There can be any number of blocks in any order.

**Decoration blocks** are groups of inactive graphical objects that are used to help convey the visual look or titles of the test fixture. The form of a Decoration block is as follows:

**BEGIN-BLOCK**

*block type ascii code*  
*number of graphical objects*  
*{list of graphical objects: one object per line}*  
(See Graphical Object ASCII Formats Section 1.3.1.3)  
**END-BLOCK**

**Connector blocks** define a group of pins which correspond to a test fixture connector. These pins are **hot spots** which the user of the test fixture can select. By convention each pin is defined by three graphical objects: A rectangle or ellipse followed by two text objects. All three of these objects are highlighted when the user of the test fixture clicks with the mouse to select the rectangle or ellipse. The first text object forms a title that is constant in the test fixture picture, the second text object forms a title which is replaced by a pin name assigned by the user. If the user de-assigns the pin, the second text object reverts to the default text in this ASCII file. The form of a Connector block is as follows:

**BEGIN-BLOCK**

*block type ascii code*  
*connector name*  
*pin type*  
*number of pins*  
*number of graphical objects*  
*{list of graphical objects: Sets of Three Objects-*  
(See Graphical Object ASCII Formats Section 3.3.1.2.1)  
*rectangle/ellipse ,*  
*fixed text label,*  
*replace-able text label}*

**END-BLOCK**

The *connector name* is the text that will appear in the fixture connector column of the **Card Assignment** grid in the Open/Create Test Fixture Dialog Box. (See Part Pin Assignment Section 3.3.2). It cues the user to the type of connector defined for this block. By convention, all pins in a connector group are of the same *pin type* and must be one of the following:

## OUTPUT-ONLY INPUT/OUTPUT

The *number of pins* designates the total pins for this connector group. The pin type along with the number of pins control which Instrument Card types will appear in the list of cards in the system card column of the **Card Assignment** grid in the Open/Create Test Fixture Dialog Box. (See Part Pin Assignment Section 1.3.2). Specifically, only compatible system cards with the same pin type and same number of pins will be available to the user for assignment to this connector group.

The *number of graphical objects* will always be three times the number of pins.

1.3.1.3 Graphical Object ASCII Formats. Graphical objects in a test fixture ASCII file are of the following types

<i>object type</i>	<i>ascii code</i>
Line	1
Rectangle	2
Ellipse	3
Text	4
Empty Rectangle	5
Empty Ellipse	6

Following is an example of lines from a test fixture ASCII file showing the definition graphical objects in a Decoration Block and a Connector Block

2 0 0 0 0 2 0 0 12632256 0 400 100 700 340	(Rectangle Object in Decoration Block)
4 0 0 0 0 16777215 550 130 730 355 0 -900 10 14 Pin DIP	(Text Object in Connector Block)
2 1 1 3 -3 1 0 0 15780008 5 368 116 400 132	(Rectangle Object in Connector Block)
4 1 0 0 0 16777215 415 116 465 132 0 0 5 Pin 1	(Text Object in Connector Block)
4 1 0 0 0 16711680 292 116 392 132 2 0 8 PIN (1)	(Text Object in Connector Block)

**Rectangle and Ellipse** objects have the following format:

```
<object type> <card pin number> <selectable> <select type> <sibling>
<pen width> <pen color>
<brush style> <brush color> <brush hatch>
<x1> <y1> <x2> <y2>
```

**Empty Rectangle and Empty Ellipse** objects have the following format:

```
<object type> <card pin number> <selectable> <select type> <sibling>
<pen width> <pen color>
<x1> <y1> <x2> <y2>
```



**Text** objects have the following format:

*<object type>* *<card pin number>* *<selectable>* *<select type>* *<sibling>* *<color>*  
*<left>* *<top>* *<right>* *<bottom>*  
*<text align>* *<rotate angle>*  
*<string length>* *<string>*

The *object type* is as defined above.

The *card pin number* specifies which pin on the instrument card this connector pin is attached to. For Decoration Blocks, card pin number is 0.

The *selectable* parameter determines whether this object is a user selectable **hot spot**. (No=0, Yes = 1) For Decoration Blocks this parameter is 0. In a Connector Block, the rectangle/ellipse object of a pin set should be set to 1, the two associated text objects should be set to 0.

The *select type* parameter is an arbitrary number assigned to a pin set by the test fixture designer. It is used when the user is selecting groups of pins to assure that the pins are all of a compatible type. For Decoration Blocks this parameter is 0.

The *sibling* parameter is used by the test fixture designer to chain together some number of pins within a given connector. When a user selects a pin with siblings, the selected pin along with all of its siblings are highlighted. The typical use of the sibling parameter is to group pins into I/O pairs. Another example is for buss connections which contain multiple siblings. Siblings are specified as circular links. In other words, the last sibling in a set points back to the first. The number given in the sibling parameter specifies a relative line number for the next sibling in this sibling set. In the above example, the sibling parameter is '-3', indicating that the next sibling in this set is three lines back in the ASCII file. For Decoration Blocks this parameter is 0. In a Connector Block, only the rectangle/ellipse object of a pin set may have a non-zero sibling number.

The *pen width* and *pen color* parameters control the pen style used to draw the object. The color is specified as a RGB four byte value defined as follows:

<i>byte</i>	<i>meaning</i>	<i>values</i>
high byte	na	0
byte 2	BLUE intensity	(0-255)
byte 1	GREEN intensity	(0-255)
low byte	Red intensity	(0-255)

The *brush color* is defined the same way as pen color above. The *brush style* and *brush hatch* parameters are used as follows:

<i>brush styles</i>	<i>values</i>
Solid	0
Hollow	1
Hatched	2
Pre-Defined Bit Maps	3..n

<i>brush hatch</i>	<i>values</i>
Horiz	0
Vert	1
LDiag	2
RDiag	3
Cross	4
DiagCross	5

The *x1*, *y1*, *x2* and *y2* parameters specify the top left and bottom right coordinates of the object in pixels. The origin of the coordinate system is the top left corner of the picture is (0,0), increasing x is to the left, increasing y is down.

The *color* parameter for text objects is defined the same way as pen color above.

The *left*, *top*, *right* and *bottom* parameters for text objects define a bounding rectangle for the text string. The text will be automatically sized to fit this rectangle.

The *text align* parameter is defined as follows:

<i>byte</i>	<i>values</i>
Left	0
Center	6
Right	2

The *rotate angle* parameter specifies a rotation angle for the text in .1 degree increments, counter clockwise from the three o'clock position. For example, a value of 450 will display text at 45 degrees.

The *string length* parameter specifies the number of characters in the *string* parameter including spaces.

1.3.1.4 Testing And Loading A Text Fixture (\*.fxt) File. After a Test Fixture ASCII files (\*.fxt) is created, it must be loaded into a data base before it can be used.

1.3.1.5 Testing a Test Fixture ASCII File. Prior to loading a test fixture, it is a good idea to error check the file. This is accomplished by selecting **File|Preview TstFxt File**. The ASCII file will be parsed and the graphic picture will be displayed. The preview halts upon errors found in the ASCII file. Errors are reported to the screen, giving the nature of the error and the

ASCII file. Errors are reported to the screen, giving the nature of the error and the line location. The test fixture picture will be drawn up to the place in the file where errors were detected.

**1.3.1.6 Loading a Test Fixture ASCII File.** Loading a Test Fixture ASCII file into a data base is accomplished by selecting **File|New|TestFixture**. The Open File Dialog Box appears. Select the drive, path and file name of the (\*.fxt) test fixture ASCII file. Select the data base that the test fixture will be loaded into. Press OK.

### 1.3.2 Part Pin Assignment.

Pin assignments of a specific part in a test fixture must be made prior to running a test. This is accomplished in a pin assignment Dialog Box which is accessed by selecting **File|New|TestFixture:PinConfig** or by selecting **File|Open|TestFixture:PinConfig**. The dialog box illustrated below in Figure 1-13 will appear.

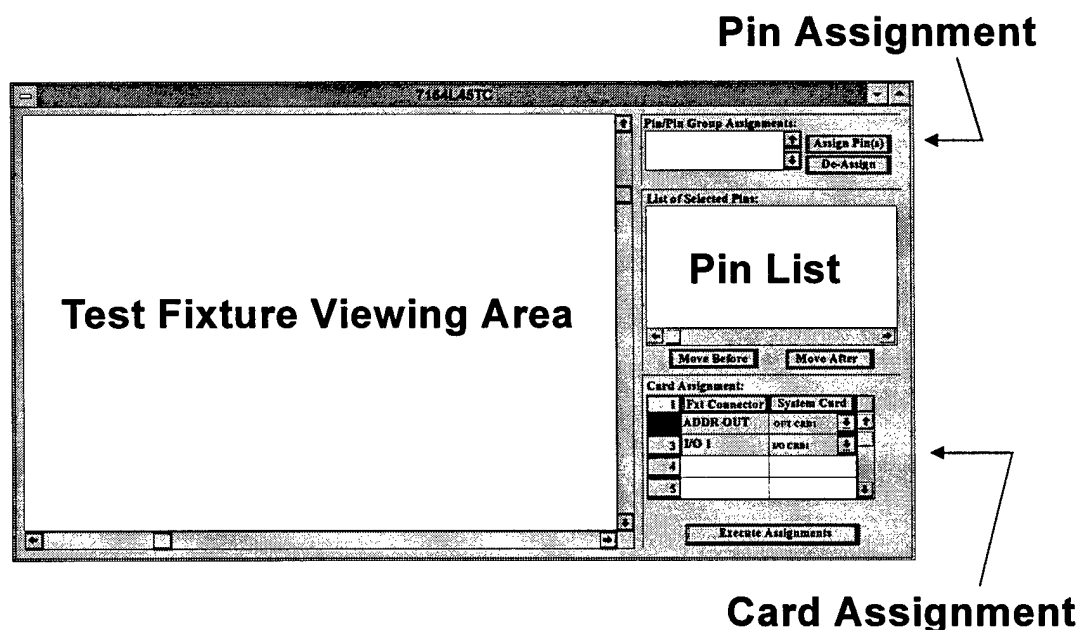


Figure 1-13. Test fixture and pin assignment.

**1.3.2.1 Test Fixture Viewing Area.** The selected test fixture is displayed in the test fixture viewing in the dialog box as shown above. Initially, the test fixture is drawn at 100% resolution. Zooming in or out to different resolutions is accomplished by selecting **File|View|**. Available resolutions are: Fit In Window, 100%, 125%, 150% and 200%.

Creating a part pin assignment is accomplished by first assigning instrument cards to the test fixture connectors, then selecting pins and groups of pins and assigning pin names.

**1.3.2.2 Test Fixture Connectors and Card Assignments.** To perform a test, loaded instrument drivers control instruments which are electrically connected to test cards. The user must specify the physical inter-connect between instruments and cards by assigning a test fixture connector to an instrument card.

A connector is a group of pins on the test fixture of all the same pin type. Connectors are determined by the designer of the test fixture. ( See Creating A Test Fixture Section 1.3.1.1 and Test Fixture ASCII File Format Section 1.3.1.3). Typically, a graphical test fixture connector is the same as a physical connector on the test card, but it does not have to be. A physical test card connector may be subdivided into several graphical connectors, or a group of physical connectors may be implemented as one graphical connector, as long as the above criteria is met.

Instrument cards are defined by the instrument driver.

When making a new pin assignment, initially the test fixture pins are grayed and the pin hot spots are turned off. To make pins selectable, connectors must first be assigned to instrument cards in the **Card Assignment** group of the dialog box shown above. The center column of the Card Assignment group lists the connectors in this test fixture. The right column contains a drop down list which displays the instrument cards that are available to assign to the selected test fixture. Available cards are dependent upon the instrument drivers loaded in this session, and of those, which cards are compatible with the connector, and have not been previously selected. Compatibility is achieved when the instrument card has the same number of pins as the test fixture connector, and all the pins are of the same type.

Also included in the drop down list, is the **Locally Enable** option. This option enables a connectors pins without assigning an instrument inter-connect. This allows for the creation of generic pin assignments whose instrument connection are specified in a later application.

**1.3.2.3 Making Pin Assignments.** Pin assignments involves naming pins and groups of pins on the test fixture to correspond to the part to be tested. These pin names are used by instrument drivers to properly stimulate the DUT. To make pin assignments, move the mouse cursor over a desired pin then press the left mouse button. The pin and its associated names become highlighted. To de-select the pin, click elsewhere in the test fixture drawing area. Multiple pins are selected by depressing the **Shift Key** while clicking on the desired set of pins.

Once the pin or set of pins have been highlighted, assign a name by pressing the **Assign Pin(s)** button. A dialog box will popup requesting the name for this pin group. Enter a name, Press OK. The pin label names will be replaced by the new pin name. This name will also be placed in the **Pin/Pin Group Assignments** list in the dialog box as shown above.

If more than one pin was selected, the pin names will include an index number starting with zero. For example, if 4 pins were selected and given the name ADDR, the resulting pin labels would be ADDR0, ADDR1, ADDR2 and ADDR3. The index sequence of pins in a group will be the same order in which they were selected. In many testing applications this sequence is critical. To change the pin sequence for an existing group, see **Modifying Pin Groups** Section 1.3.2.5.

1.3.2.4 Pin List. As you select pins, the details information is shown in the **List of Selected Pins** group of the dialog box shown above. The details are:

User Assigned Name For the Pin (if one exists yet)  
Test Fixture Name For the Pin  
Test Fixture Connector this Pin Belongs to  
Instrument Card This Connector is Assigned to

Also, a the pin list can be displayed for an existing pin/pin group by double clicking the pin/pin group name in the Pin/Pin Group Assignments list.

1.3.2.5 Modifying Pin Groups. The sequence of pins for an existing pin group can be modified from the **List of Selected Pins** group of the dialog box. This is accomplished by highlighting the desired pin to move in the sequence, then selecting either the **Move Before** or **Move After** buttons, and then double clicking on the pin list the name of the pin which the subsequent pin is to be moved before/after.

1.3.2.6 Printing Pin Lists And Test Fixtures. You may make a hard copy record of the pin list along with the test fixture illustration by selecting **File|Print**. A dialog box will appear allowing you to selecting printing organized by pin group or by test fixture connector. You may additionally select whether or not to print the test fixture graphics. You may preview the pin list hard copy by selecting **File|Print Preview**.

### 1.3.3 Making Timing Diagrams.

A **Timing Diagram** is used to specify the timing for stimulation of the DUT. Each ATV pin has independent timing, and can store up to 7 timing sets which are selectable on-the-fly. Timing is specified by graphically creating timing diagrams similar to the ones found in part specification sheets.

Timing Diagrams are created in a dialog box which is accessed by selecting **Instruments|ATV|Timing Diagram|New** or by selecting **Instruments|ATV|Timing Diagram|Open**. The dialog box illustrated below in Figure 1-14 will appear.

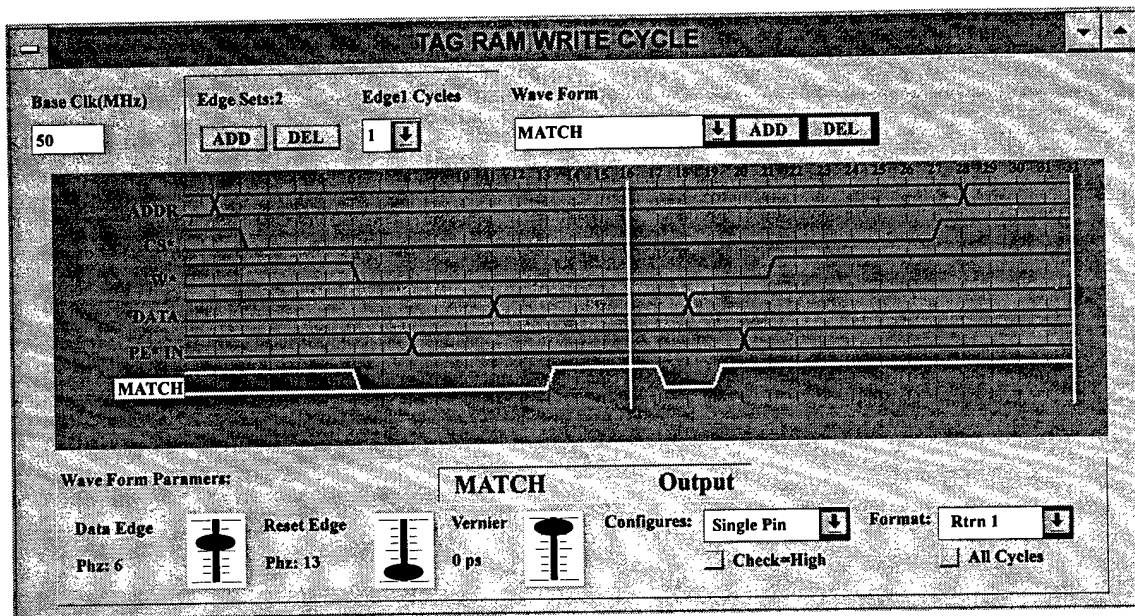


Figure 1-14. Timing diagram dialog box.

Testing involves outputting data or states to the part and reading data from the part in a specific timing. Output and compare data is specified in a test algorithms . Data can be generated algorithmically or can come from test vector files. (See Creating an Algorithm Section 1.3.5). Timing is specified in timing diagrams. A timing diagram is composed of **waveforms**. Waveform are given names which are matched to pin names from the part/package description to configure ATV pins. (See Test Fixtures Section 1.3.1).

A new waveform and its name are created by using the mouse to depress the **ADD** button shown in the above dialog box. A popup dialog box appears prompting for the name of the waveform. The popup also contains a list box where you specify the **Pin Type** as **Output** or **Cmpr**. Select an output pin type if this waveform is to assert data, or compare if it specifies the timing for which data will be read from the part. The new waveform is inserted directly below the waveform presently highlighted. A waveform can be highlighted by placing the mouse cursor over the waveform illustration and pressing the left mouse button, or by selecting the waveform name from the list control in the WaveForm box. A waveform can be deleted by using the mouse to depress the **DEL** button.

Waveforms configure either **Single Pins** or **Pin Groups**. This is specified in the **Configures** list control of the **WaveForm Parameters** box shown above. When configuring single pins, the

output data can optionally be specified in the timing diagram instead of an algorithm. Single pin output state is specified in the check box under the Configures control. (**Check=High**).

A waveform typically has a leading edge, called the **Data Edge**, and a trailing edge called the **Reset Edge**. A data and reset edge set define the timing in which data is output or compared. A timing diagram may contain as many as seven **Edge Sets**.

(Note: A test may be composed of multiple timing diagrams, and can incorporate timing diagrams until seven edge sets are declared. For example, if a test uses a timing diagram that contains 5 edge sets, two edge sets remain for other timing diagram declarations. See Creating an Algorithm Section 3.3.5).

The number of edge sets is increased or decreased by using the mouse to depress the ADD and DEL buttons in the Edge Sets portion of the above dialog box. Edge set boundaries are illustrated with bold vertical white lines in the timing diagram.

Timing formats determine the state of the pin(s) at the completion of the data cycle. Timing formats include Return to One (**Rtrn 1**), Return to Zero (**Rtrn 0**), Return to Complement (**Rtrn C**), **NRZ**, or **No Change**. Formats are specified in the **Format** list control of the **WaveForm Parameters** box shown above. Formats are defined as follows:

Rtrn 1 -	Output goes to data value on data edge, goes to output high on reset edge
Rtrn 0 -	Output goes to data value on data edge, goes to output low on reset edge
Rtrn C -	Output goes to data value on data edge, goes to compliment of data value on reset edge
NRZ -	Output goes to data value on data edge and stays there, no reset edge
No Change -	No edges, values remain at ending state of previously applied timing Diagram

The minimum **cycle** time for a timing diagram is set by the test frequency and can be as short as 20 ns. Each edge set can have a duration of 1 to 7 cycles. The maximum number of cycles definable in a timing diagram is 7 edge sets times 7 cycles each, for a total of 49 cycles.

**Rough edge placement** is 1/16 cycle boundaries. This gives down to 1.25 ns independent edge placement. Edges can be placed using either the slider controls or by dragging an edge using the mouse.

**Fine edge placement** is 100 ps resolution and is set with the **Vernier** slider control. Vernier moves both the data and reset edge together.

**Blanking Pulse** - ATV disallows an edge to be placed in the first phase of the first cycle of a timing diagram due to a hardware imposed blanking pulse.

Slider controls also respond to the up and down arrows for fine control. To use the up and down arrows, first left mouse click on the slider which you wish the arrows to control.

Vernier value is reported to the left of the slider control in actual time with units of picoseconds.

Edge timing is reported to the left of the slider control. By default, edge placement is reported in terms of cycle phase. You may optionally view edge placement in terms of **pseudo-time** by selecting **View|Change Units**. This causes edge placement to be reported in units of nanoseconds. Actual test frequency is specified at run time. Values presented next to the *Edge* slider controls are *pseudo-times* and given simply as an aid to timing diagram creation. The pseudo-times are based on the cycle frequency currently specified in the **Base Clk** control illustrated in the above. Edge placement time is the sum of edge value plus the vernier value.

**Zoom** - You may magnify a portion of the timing diagram for fine edge placement or reading by selecting **View|ZOOM IN**. This caused the cursor to change to a magnifying glass shape when the mouse is over the timing diagram drawing area. Depress the left mouse button and drag left or right to create a zoom window. After the releasing the left mouse button, the timing diagram will redraw at the selected resolution. To zoom out, select **View|ZOOM OUT**.

**All Cycle** - If a timing diagram contains an edge set that has more than one cycle, a waveform may be optionally repeated for every cycle as long as the data and reset edge occur within a cycle boundary. This can be helpful in certain applications such as constructing clocks. This is specified in the All Cycles check box control as illustrated above.

#### 1.3.4 Algorithm Project.

An algorithm project combines a test algorithm with a test fixture and timing diagrams to create a test. It is also where algorithm source code is created and compiled and where download files are built for ATV. Algorithm Projects are accessed by selecting **Instruments|ATV|Timing Diagram|New** or by selecting **Instruments|ATV|Timing Diagram|Open**. The dialog box illustrated below in Figure 1-15 will appear.

An algorithm project builds a group of three download files all with the same name but different extensions (\*.AXT, \*.AXF, \*.AXG) defined as follows:

- \*.AXT - Opcode file containing algorithmic binary code for the ATV processor.
- \*.AXF - Timing code file for output pins.
- \*.AXG - Pin grouping and type information file.

The file name is indicated in the **Executable Names** text box at the top of the algorithm dialog box. These files are used to configure the ATV for test at run time. (See Test Execution 1.3.6).

The algorithm project dialog box is subdivided into Source Code, Text Fixture and Timing Diagram groups. The following presents how to use controls in each of these areas. Detailed information on creating algorithms and building download files is given in Creating an Algorithm.



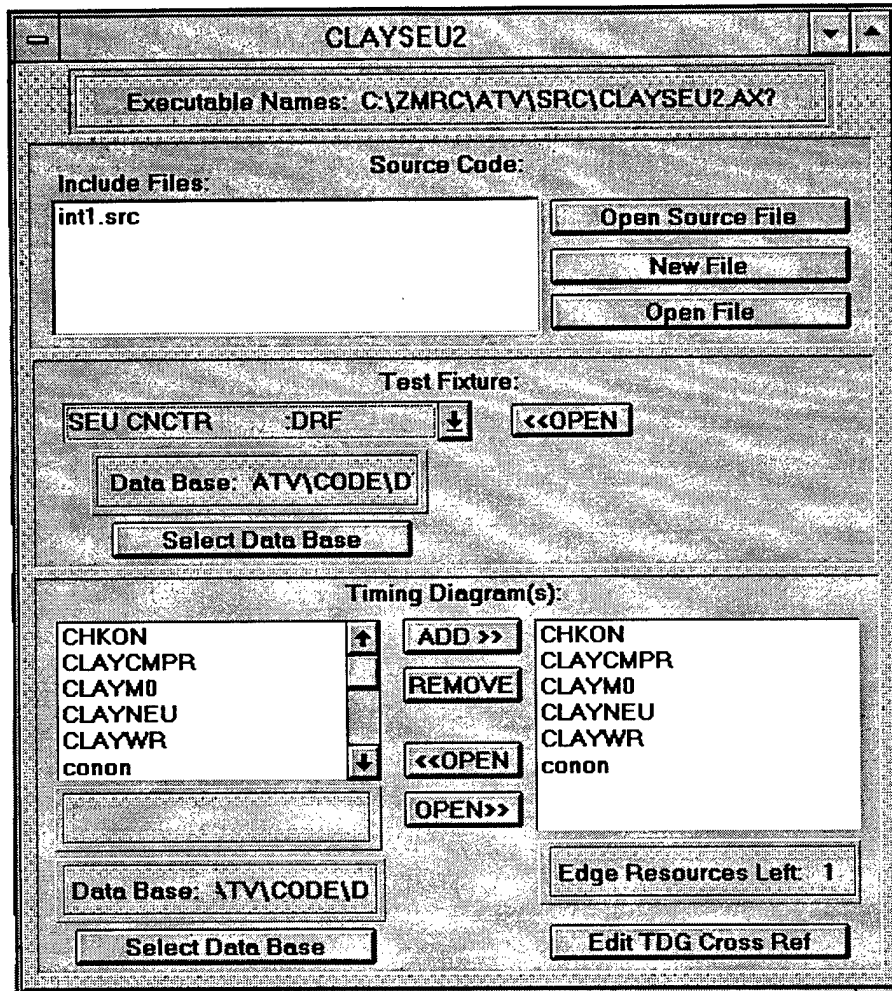


Figure 1-15. Algorithm project dialog box.

1.3.4.1 **Source Code.** Source code for the algorithm is from a file with the executable name and a \*.SRC extension. It can be accessed for editing automatically by using the mouse to depress the **Open Source File** button shown in the above dialog box. Any include files declared in the source file are listed in the **Include File** list box control. Include files can be automatically opened by double clicking on the include file name. The algorithm project also allows for the creating and opening of any ASCII text file by using the **New File** and **Open File** buttons.

1.3.4.2 **Test Fixture.** The test fixture to be used with this algorithm is selected in the **Test Fixture** group. The drop down list presents all the test fixtures available from the current data base. Use the **Select Data Base** button to view test fixtures from an alternate data base. Select a test fixture to be used for this algorithm by highlighting it's name in the drop down list. View the test fixture by using the mouse to depress the **<<OPEN** button.

1.3.4.3 Timing Diagram(s). The timing diagrams to be used along with this algorithm are selected in the **Timing Diagram(s)** group. The left panel list control presents the timing diagrams available from the current data base. Use the **Select Data Base** button to view test fixtures from an alternate data base. Select timing diagrams to attach to this project by using the **ADD>>** button. An algorithm test can have up to 7 edge set resources. (See Making Timing Diagrams Section 1.3.3). As timing diagrams are added to this project, the remaining available resources are reported in the **Edge Resources Left** box.

The algorithm source code declares place holders for some number of timing diagrams. Timing diagrams specified in this project are linked into these place holders when the ATV download files are built. (See Creating an Algorithm Section 1.3.5). By default the name used in the algorithm is expected to be the name shown in the timing diagram list. To use timing diagrams with names that do not match the algorithm declarations, make a timing diagram cross reference by using the mouse to depress the **Edit TDG Cross Ref** button.

### 1.3.5 Creating an Algorithm.

Creating an algorithm involves writing source code, compiling it with timing diagrams selected in the algorithm project, then building ATV download files. This section describes the details of how to create a new algorithm.

Create a new Algorithm Project by selecting **Instruments|ATV|Timing Diagram|New**. The algorithm project dialog box appears. (See Algorithm Project Section 1.3.4). Use the mouse to depress the **Open Source File** button. A dialog box appears prompting for the **Source Code Base Directory** for this project and the name for the **Executable File Name** set. The algorithm project creates four files with the same name and different extensions. (See Algorithm Project Section 1.3.4). The executable files will be placed in the base directory. To have the files placed in the current directory leave the base directory blank. Enter the path, and an executable name with no extension and press Enter. The text editor appears with template source code as shown below in Figure 1-16:

All ATV algorithms must include the '**interrupt.src**' file which sets up the interrupt routines for the processor. A default **interrupt.src** is shipped along with ATV. Move a copy of **interrupt.src** to the base directory for this project.

Next you will write and debug algorithm source code. Language elements and examples are given in the Programmers Guide. This example will use the code presented below in Figure 1-17 to complete the discussion of algorithm creation.

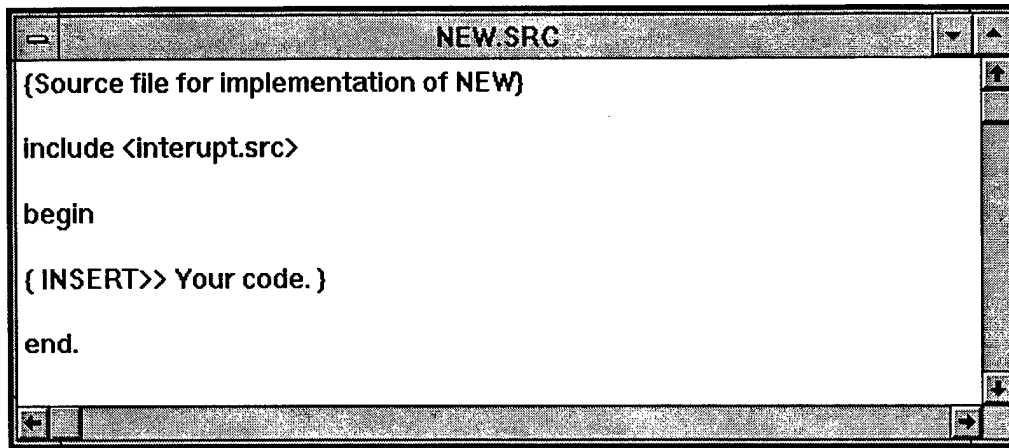


Figure 1-16. Text editor with ATV algorithm template code.

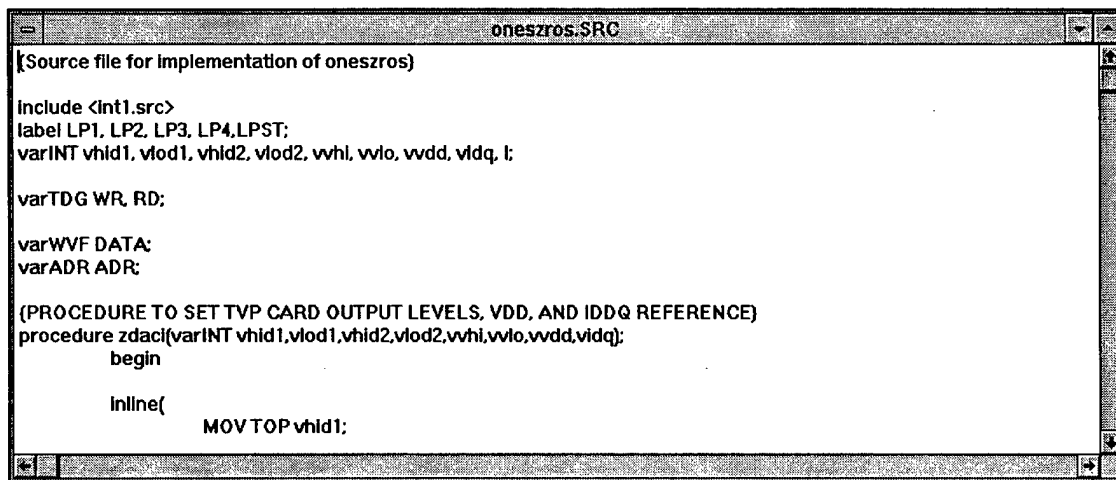


Figure 1-17. Sample algorithm code.

Select timing diagrams appropriate for your algorithm as described in Algorithm Project. Timing diagram selection is dependent upon declarations in the source code. The source code declares **varTDG** variables which are place holders for timing diagrams. In the above example two such variables are shown; WR and RD, indicating that two diagram are needed for this project. Also the source code declares **varWVF** and/or **varADR** waveform variables for the selected timing diagrams. In this example the DATA and ADR waveforms of the WR and RD timing diagrams will be used. Therefore, two timing diagrams must be selected for this example project with the names WR and RD and they must include waveforms named DATA and ADR. (Note: The timing diagrams can have different names than those declared with varTDG if a cross reference is used. See Algorithm Project Section 1.3.4).

As the code is being written, compile and debug it by selecting **Algorithm|Compile**. A dialog box appears reporting the status of compilation. If errors exist an **Error Message** box appears. Double clicking on an error in the Error Message Box causes the line containing the error in the source code to be highlighted.

It is possible to view the assembly code that is resulting from the compilation by selecting **View|SetView**. A dialog box appears allowing for the selection of viewing assembly after PAS1, PAS2 or Link, and for viewing Mixed Assembly and source code or assembly only. When the assembly code is viewed, it is also placed in a \*.ASM file with the executable name for subsequent viewing.

After the algorithm successfully compiles, the next step is to build the executable file set for down load to the ATV processor. First select the test fixture that supplies pin names as described in Algorithm Project . Next select **Algorithm|Build DownLoad**.

The algorithm is now read for Test Execution.

#### 1.3.6 Test Execution.

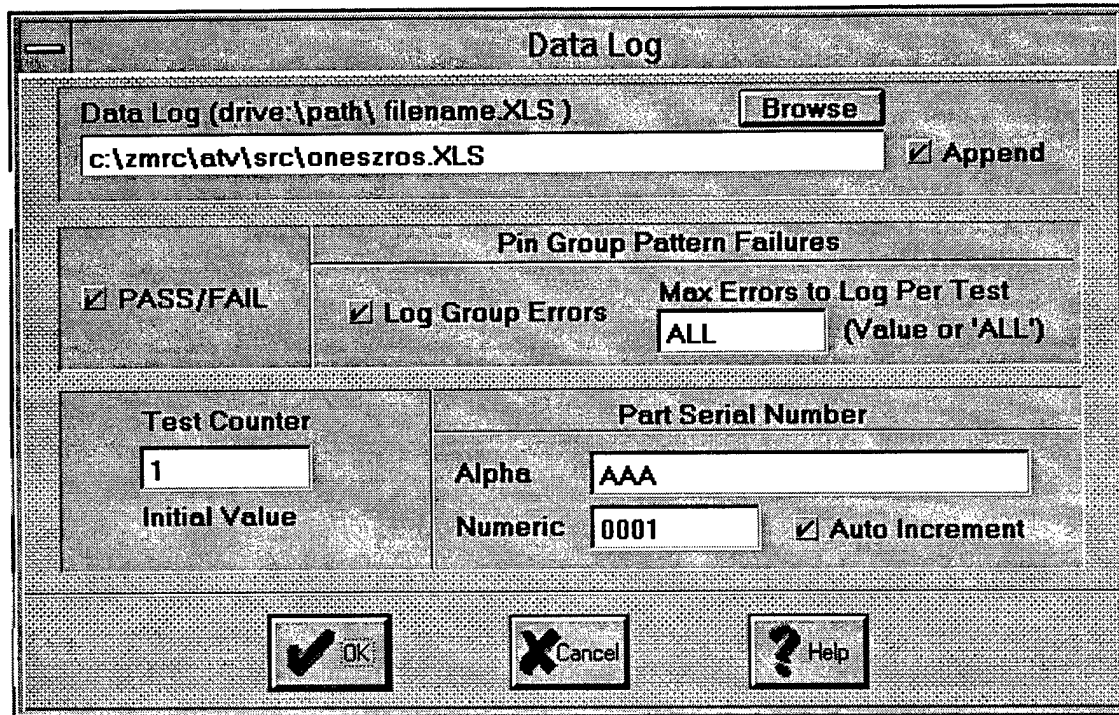
1.3.6.1 Data Log Setup. Test results are placed in EXCEL style spread sheet. Prior to running a test, setup up data logging options by selecting **Algorithm|DataLog Setup**. The Data Log dialog shown below in Figure 1-18 will appear.

By default, results will be placed in an XLS spreadsheet with the same name and path as the executable. You may select a different file by editing the contents of the Data Log edit box.

If the **Append** box is checked, data will be added to existing data in the named file on consecutive runs, otherwise data will be overwritten.

Checking the **PASS/FAIL** box will cause ATV to report a pass or fail result in the data log for each test. Checking the **Log Group Errors** box will cause ATV to report detailed pin failure information. The **Max Errors to Log Per Test** edit control specifies a value after which error reporting is truncated.

**Test Counter Initial Value** controls the initial test number reported in the data log. **Part Serial Number** is composed of an alpha and a numeric string. By checking the **Auto Increment** box the numeric portion will be automatically incremented on subsequent tests.



The image shows a 'Data Log' dialog box with the following fields and controls:

- Data Log (drive:\path\ filename.XLS)**: A text field containing 'c:\zmrclatv\src\oneszros.XLS' and a **Browse** button.
- Append**: A checked checkbox.
- PASS/FAIL**: A checked checkbox.
- Pin Group Pattern Failures**: A section containing:
  - Log Group Errors**: A checked checkbox.
  - Max Errors to Log Per Test**: A text field containing 'ALL' with the note '(Value or 'ALL')'.
- Test Counter**: A text field containing '1' with the label 'Initial Value' below it.
- Part Serial Number**: A section containing:
  - Alpha**: A text field containing 'AAA'.
  - Numeric**: A text field containing '0001'.
  - Auto Increment**: A checked checkbox.
- Buttons**: **OK** (with a checkmark icon), **Cancel** (with an 'X' icon), and **Help** (with a question mark icon).

Figure 1-18. Data log setup.

1.3.6.2 Loading and Running the Algorithm Test. Prior to running an algorithm test the executable files must first be downloaded to the ATV processor. To load the test select **Algorithm|Load**. Next run the test by selecting **Algorithm|Run**. The **Run Algorithm** dialog box shown below in Figure 1-19 appears.

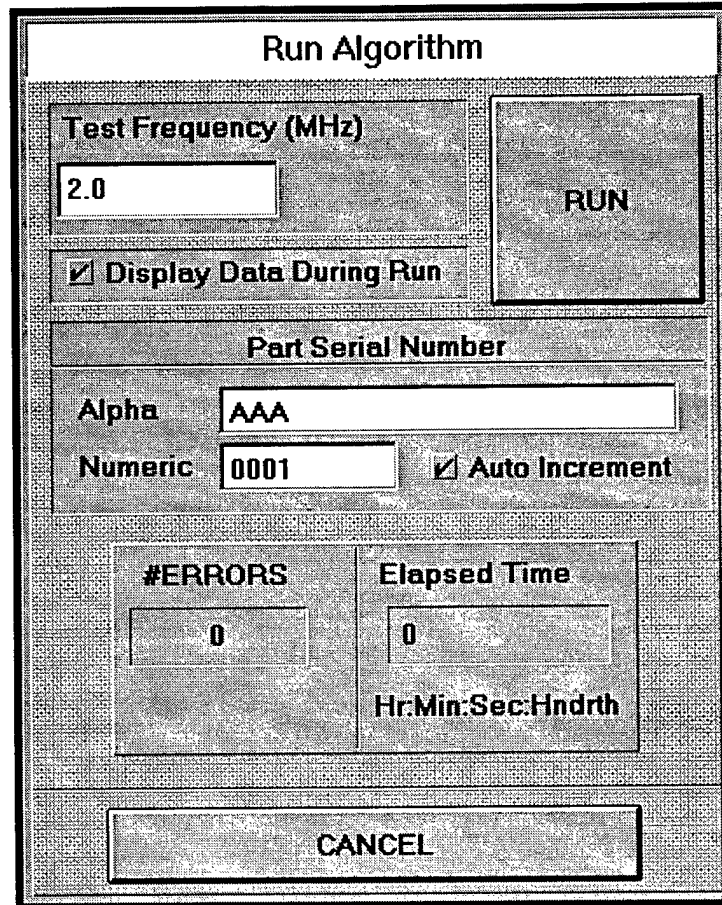
Specify the **Test Frequency** in units of MHz. The Run Algorithm dialog box allows the Part Serial Number to be specified and modified between tests. It is composed of an alpha and a numeric string. By checking the **Auto Increment** box the numeric portion is automatically incremented on subsequent tests.

The data log is initially minimized as an icon in the lower left portion of the screen. To view data during test execution, double click on the data log icon.

Use the mouse to depress the **RUN** button to initiate testing. **#ERRORS** and Elapsed time is reported in the dialog box.

Tests may be run repeatably using the RUN button.

To exit the run algorithm mode, use the mouse to depress the CANCEL button.



The image shows a 'Run Algorithm' dialog box. It has a title bar 'Run Algorithm'. Inside, there's a section for 'Test Frequency (MHz)' with a text box containing '2.0'. To the right of this is a large 'RUN' button. Below the frequency section is a checkbox labeled 'Display Data During Run' which is checked. Below that is a section titled 'Part Serial Number'. It contains two text boxes: 'Alpha' with 'AAA' and 'Numeric' with '0001'. To the right of these is another checked checkbox labeled 'Auto Increment'. At the bottom of the dialog, there are two small display boxes: '#ERRORS' showing '0' and 'Elapsed Time' showing '0'. Below these is a label 'Hr:Min:Sec:Hndrth'. At the very bottom is a large 'CANCEL' button.

Figure 1-19. Run algorithm dialog box.

### 1.3.7 Data Log.

A sample ATV data log XLS file is shown below in Figure 1-20.

In this example, full pin error reporting was turned on. The **PC** column gives the hexadecimal address line from the algorithm source code where the part experienced failure. The PC number can be used in conjunction with the \*.ASM file generated during compile. (See Creating an Algorithm Section 1.3.5).

Pin groups as defined in the Test Fixture are displayed in subsequent columns. If the pin group is of input/output type, two columns are generated; one for expected data pattern (**EXP PTRN**), one for the error pattern (**ERR PTRN**). The error pattern presents an 'X' for any failing bit. Bit order for the group is as defined in the test fixture.

NATL_RAM.XLS									
	A	B	C	D	E	F	G	H	I
1		TimeStamp	PC	DATA	DATA	ADR	WE <sup>A</sup>	WE <sup>A</sup>	OE <sup>A</sup>
2				EXP PTRN	ERR PTRN		EXP PTRN	ERR PTRN	EXP PTRN
3		11:00:11.33 1/4/1980	170	00110011	0000X000	0x0	0	0	0
4			170	00110011	0000X000	0x1	0	0	0
5			170	00110011	0000X000	0x2	0	0	0
6			170	00110011	0000X000	0x3	0	0	0
7			170	00110011	0000X000	0x4	0	0	0
8			170	00110011	0000X000	0x5	0	0	0
9			170	00110011	0000X000	0x6	0	0	0
10			170	00110011	0000X000	0x7	0	0	0
11			170	00110011	0000X000	0x8	0	0	0
12			170	00110011	0000X000	0x9	0	0	0
13			170	00110011	0000X000	0xa	0	0	0
14			170	00110011	0000X000	0xb	0	0	0
15			170	00110011	0000X000	0xc	0	0	0
16			170	00110011	0000X000	0xd	0	0	0

Figure 1-20. Sample data log.

## SECTION 2 PROGRAMMERS GUIDE

### LANGUAGE STRUCTURE

ATV achieves its high performance and flexibility in large part because of the close coupling between its hardware and software. The Test Vector Processor (TVP) is a custom designed RISC processor with special features for maximizing the speed of test execution. Because of its unique design, a special set of assembly language instructions was developed for programming the processor. A custom compiler processes the assembly instructions and produces the appropriate operational codes (op codes) to be downloaded into the TVP memory.

A custom high level language was also developed which is a subset of PASCAL. This language incorporates all major aspects of modern programming languages, including procedures with parameter passing, basic branching operators, conditionals and math expressions. The language also implements an include operator which allows source code from other modules (documents) to be linked at compile time. Therefore, algorithm and language libraries can be built by MRC, or the user, to extend the language and provide basic device test functions such as GALPAT, Marching 1's, etc. The language is described in Figures 2-1 through 2-4 as syntax flow charts.

Mixed level programs can be written through the use of an 'inline' operator within the high level language. (i.e. the high level language allows inline assembly).

The compiler has comprehensive error checking for lexical (typographical), syntactical and semantic type errors. The compiler can be optionally set to display the intermediate code which is generated from the high level source code. This can be helpful to the user in optimizing algorithms. For example, viewing assembly level might be used to optimize code so test devices are exercised at the highest possible speeds by minimizing processing steps between outputs to the device.





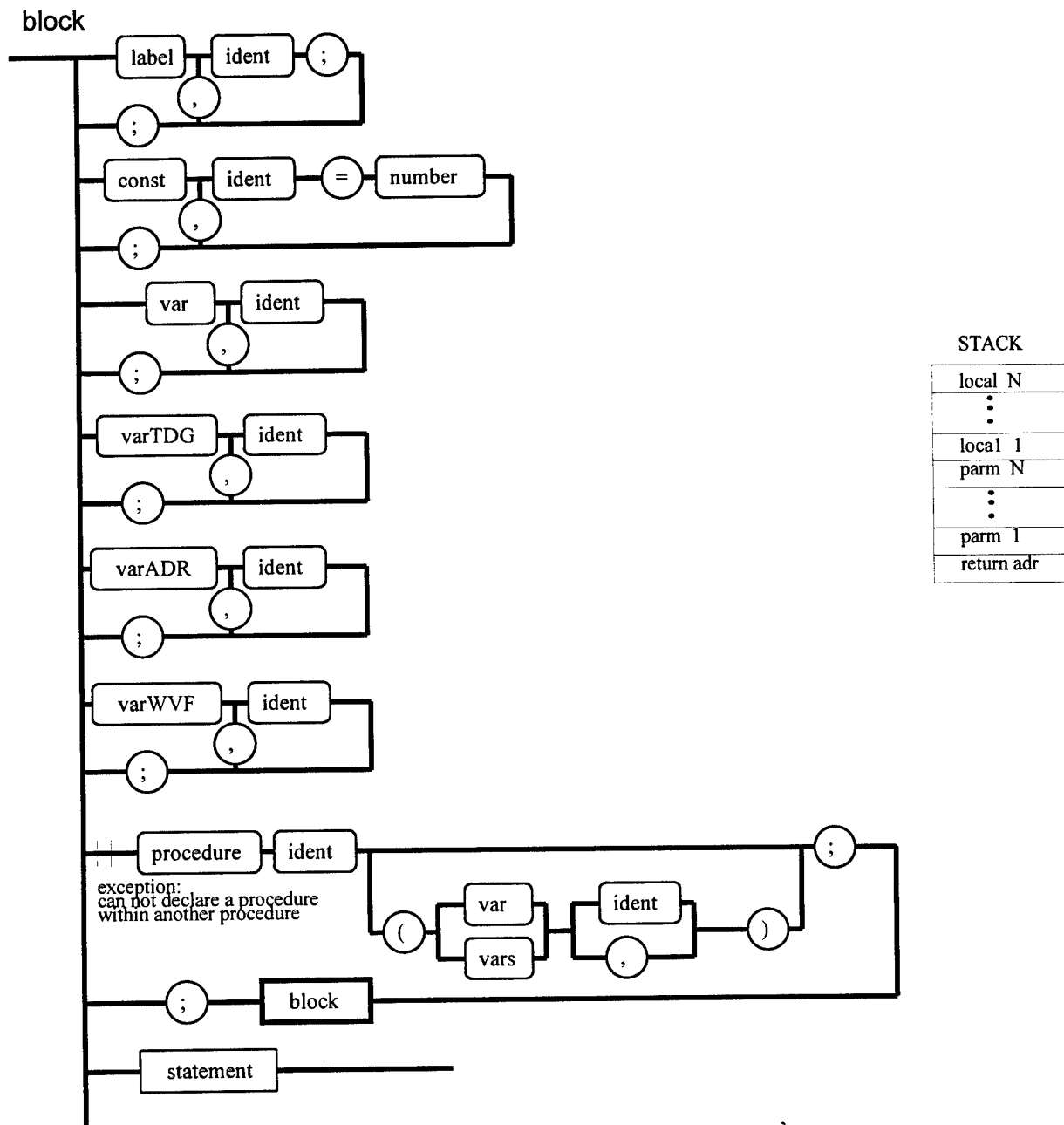


Figure 2-2. Syntax chart of a program block.

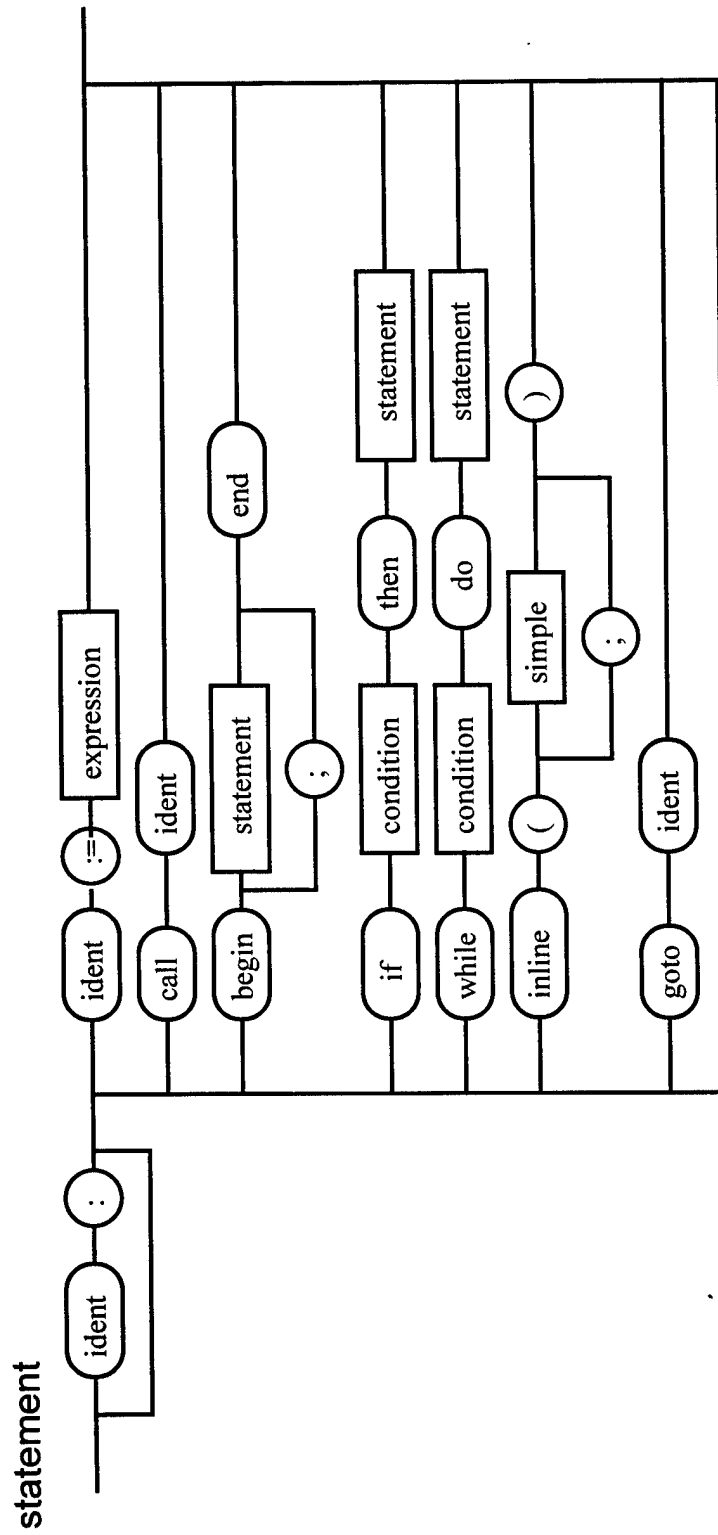


Figure 2-3. Syntax chart for high level instructions.

simple

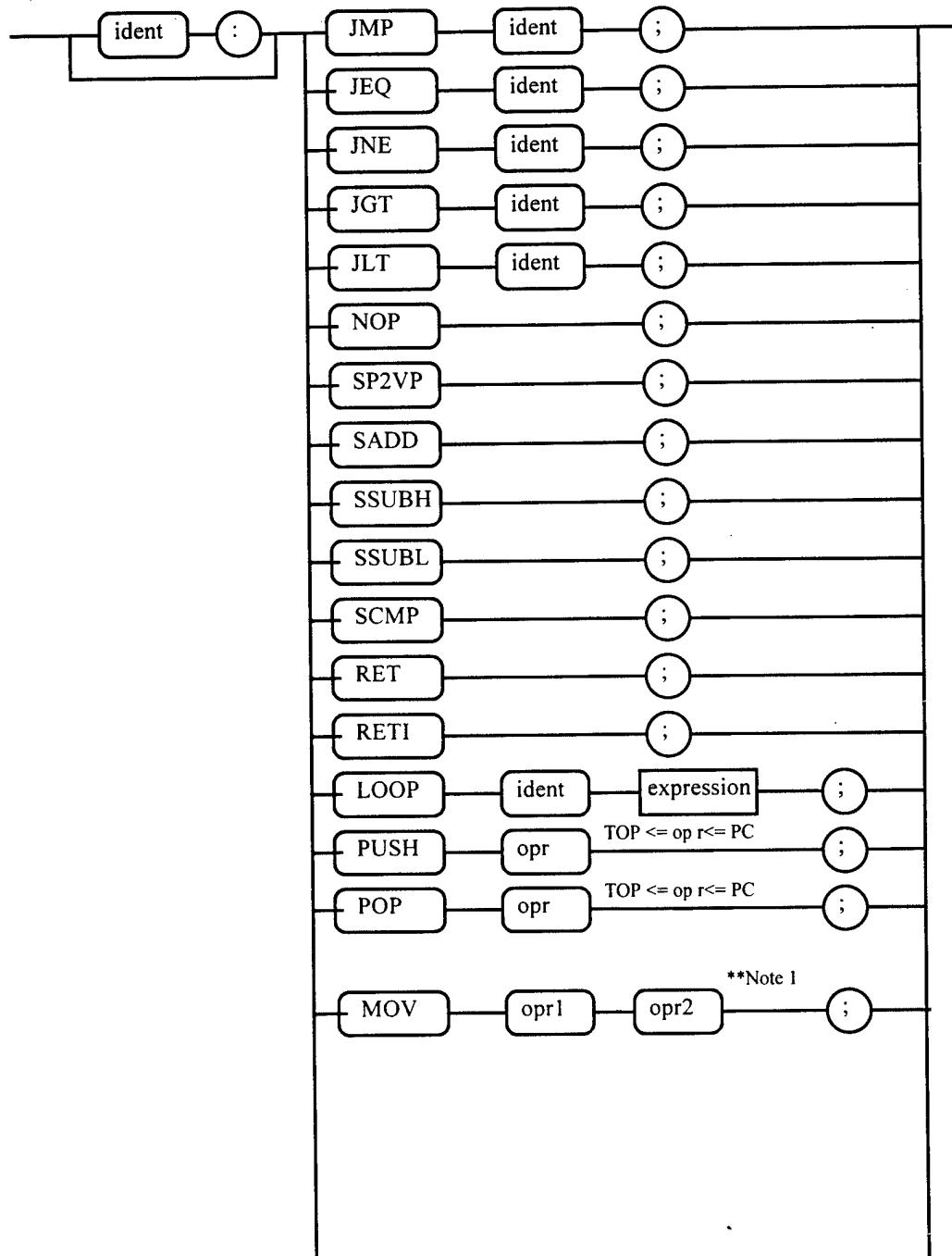
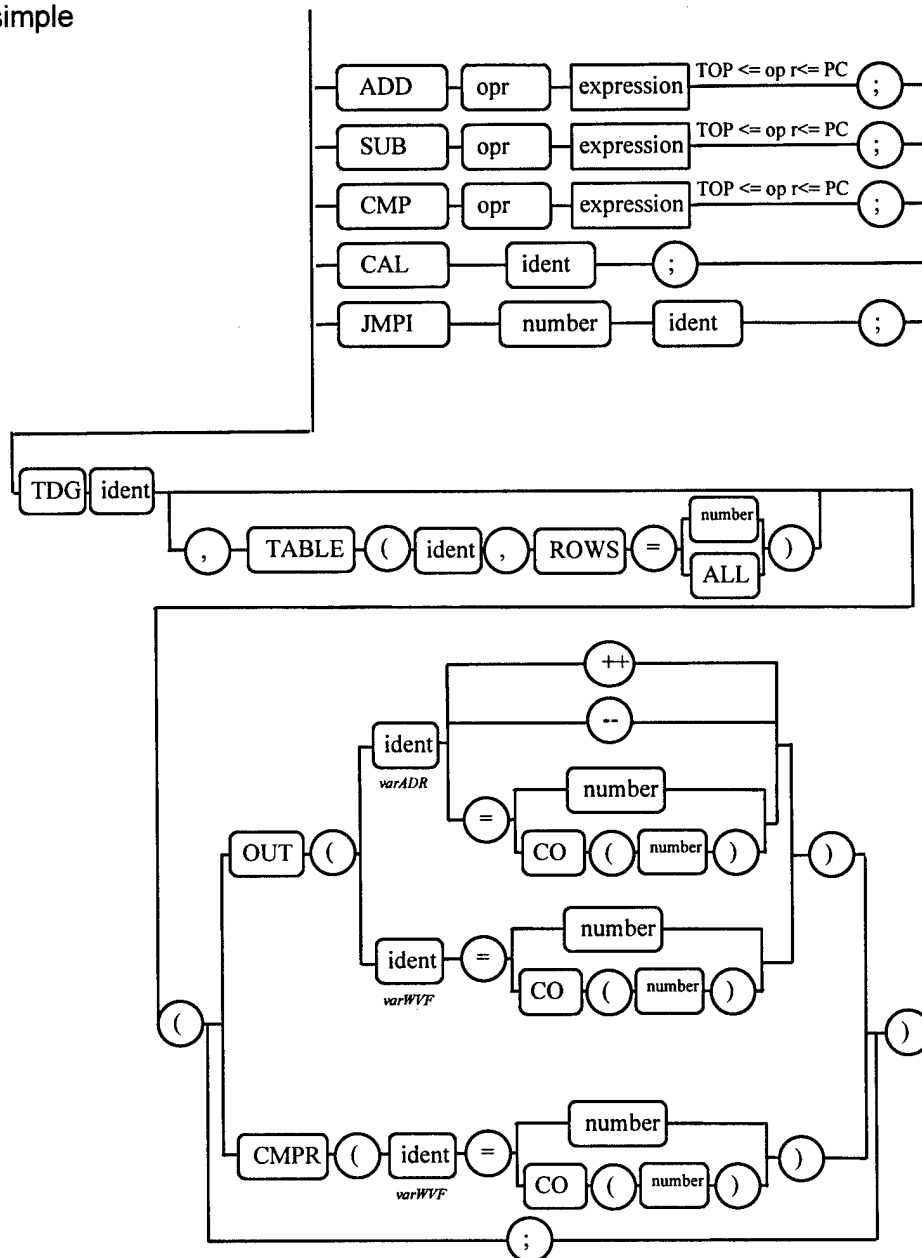


Figure 2-4. Syntax chart for assembly level instructions.

simple



opr = {[VP], TOP, BOT, EA, SA, CX, ERCNT, VP, AX, PC, SP, CR}

**\*\*Note 1:** if opr1 = [VP] then TOP <= opr2 <= PC  
if TOP <= opr1 <= AX then opr2 = { [VP], number, expression }

Figure 2-4. Syntax chart for assembly level instructions. (continued)

## **DISTRIBUTION LIST**

**DSWA-TR-97-15**

### **DEPARTMENT OF DEFENSE**

#### **DEFENSE INTELLIGENCE AGENCY**

ATTN: DT-1BJ

ATTN: TWJ

#### **DEFENSE SPECIAL WEAPONS AGENCY**

ATTN: ESA, MAJ SNOW

ATTN: ESA, W SUMMA

ATTN: ESE, L COHN

ATTN: ESE, L PALKUTI

ATTN: ESE, R C WEBB

2 CY ATTN: TRC

#### **DEFENSE TECHNICAL INFORMATION CENTER**

2 CY ATTN: DTIC/OCF

#### **FC DEFENSE SPECIAL WEAPONS AGENCY**

ATTN: FCINI

ATTN: FCTO

ATTN: FCTT, DR BALADI

#### **TECHNICAL RESOURCES CENTER**

ATTN: JNGO

### **DEPARTMENT OF THE ARMY**

#### **ADVANCED RESEARCH PROJECT AGENCY**

ATTN: ASST DIR

#### **ARMY RESEARCH LABORATORIES**

ATTN: AMSRL-PS-PD

ATTN: ANSRL-WT-NJ

ATTN: DR TIM OLDHAM

#### **ARMY SPACE & STRATEGIC DEFENSE COMMAND**

ATTN: CSSD-TC-SR

#### **DIRECTORATE FOR APPLIED TECHNOLOGY, TEST AND SIMULATION**

ATTN: ALFREDO RAMIREZ

#### **U S ARMY RESEARACH OFFICE**

ATTN: R GRIFFITH

#### **US ARMY THADD PROJECT OFFICE**

ATTN: CSSD-WD

### **USAISC**

ATTN: ASOP-DO-TL

### **DEPARTMENT OF THE NAVY**

#### **NAVAL RESEARCH LABORATORY**

ATTN: CODE 6011, C MARSHALL

ATTN: CODE 6612, D BROWN

ATTN: CODE 6613, A B CAMPBELL

ATTN: CODE 6813, N SAKS

ATTN: CODE 6816, H HUGHES

#### **NAVAL WEAPONS SUPPORT CENTER**

ATTN: CODE 6054, D PLATTETER

#### **OFFICE OF NAVAL INTELIGENCE**

ATTN: LIBRARY

#### **PROGRAM EXECUTIVE OFFICE**

#### **NAVAL AIR SYSTEMS COMMAND**

ATTN: AIR-536T

### **DEPARTMENT OF THE AIR FORCE**

#### **AIR FORCE CTR FOR STUDIES & ANALYSIS**

ATTN: AFSAA/SAI, RM 1D363

THE PENTAGON

#### **AIR UNIVERSITY LIBRARY**

ATTN: AUL - LSE

#### **PHILLIPS LABORATORY**

ATTN: CAPT CHARLES BROTHERS

ATTN: PL/VTE

ATTN: PL/VTEE, S SAMPSON

ATTN: PL/WSC

#### **ROME LABORATORIES/CC**

ATTN: ESR

#### **SMC/MCX**

ATTN: LT DAN PHILLIPS

#### **SMC/MTAX**

ATTN: K BASANY

#### **USAF ROME LABORATORY TECHNICAL LIBRARY**

ATTN: RBR

DSWA-TR-97-15 (DL CONTINUED)

WL/MTE

ATTN: MTE

**DEPARTMENT OF ENERGY**

ALBUQUERQUE OPERATIONS OFFICE

ATTN: NESD

LAWRENCE LIVERMORE NATIONAL LAB

ATTN: L - 84, G POMYKAL

ATTN: W ORVIS

LOS ALAMOS NATIONAL LABORATORY

ATTN: E LEONARD

SANDIA NATIONAL LABORATORIES

ATTN: F SEXTON

ATTN: L D POSEY

ATTN: P WINOKUR

ATTN: T A DELLIN

**OTHER GOVERNMENT**

CENTRAL INTELLIGENCE AGENCY

ATTN: OSWR/STD/MTD 5509 NHB

NASA

ATTN: CODE 900, E STASSINOPOULOUS

ATTN: K LABEL

**DEPARTMENT OF DEFENSE CONTRACTORS**

ALLIED-SIGNAL, INC.

ATTN: DOCUMENT CONTROL

ANALYTIC SERVICES, INC. (ANSER)

ATTN: A SHOSTAK

BOEING CO

ATTN: D EGELKROUT

BOOZ ALLEN & HAMILTON INC

ATTN: D VINCENT

ATTN: L ALBRIGHT

CALIFORNIA INSTITUTE OF TECHNOLOGY

ATTN: C BARNES

CHARLES STARK DRAPER LAB, INC.

ATTN: J BOYLE

CLEMSON UNIVERSITY

ATTN: P J MCNULTY

COMPUTER PRODUCTS A DIVISION OF AMPEX

ATTN: B RICHARD

ATTN: K WRIGHT

DAVID SARNOFF RESEARCH CENTER, INC

ATTN: R SMELTZER

DEFENSE GROUP, INC

ATTN: ROBERT POLL

EATON CORP.

ATTN: R BRYANT

GENERAL ELECTRIC CO.

ATTN: B FLAHERTY

ATTN: L HAUGE

GEORGE WASHINGTON UNIVERSITY

ATTN: A FRIEDMAN

HARRIS CORPORATION

ATTN: E YOST

ATTN: W ABARE

HONEYWELL, INC

ATTN: C SANDSTROM

HONEYWELL, INC.

ATTN: MS 725-5

HUGHES AIRCRAFT COMPANY

ATTN: E KUBO/S4/X301

IBM CORP.

ATTN: A SADANA

INSTITUTE FOR DEFENSE ANALYSES

ATTN: TECH INFO SERVICES

JAYCOR

ATTN: D WALTERS

JAYCOR

ATTN: CYRUS P KNOWLES

JAYCOR

ATTN: JERRY I LUBELL

JOHNS HOPKINS UNIVERSITY  
ATTN: R MAURER

KAMAN SCIENCES CORPORATION  
ATTN: DASIAC  
ATTN: DASIAC/DARE  
ATTN: R RUTHERFORD

KEARFOTT GUIDANCE AND NAVIGATION CORP.  
ATTN: J D BRINKMAN, MC 12B74

LITTON SYSTEMS INC  
ATTN: F MOTTER

LOCKHEED MARTIN CORPORATION  
ATTN: G LUM, ORG 81 - 40  
ATTN: J CAYOT, DEPT 81 - 63

LOCKHEED MARTIN CORPORATION  
ATTN: BRAIN G CARRIGG

LOCKHEED MARTIN FEDERAL SYSTEMS, INC  
ATTN: L ROCKETT  
ATTN: N HADDAD

LOCKHEED MARTIN VOUGHT SYSTEMS  
2 CY ATTN: LIBRARY EM-08

LOGICON R AND D ASSOCIATES  
ATTN: D CARLSON

MARTIN MARIETTA  
ATTN: J MILLER

MARTIN MARIETTA DENVER AEROSPACE  
ATTN: RESEARCH LIBRARY

MARYLAND UNIVERSITY OF  
ATTN: H C LIN

MAXWELL FEDERAL DIVISION, INC.  
ATTN: DR JASON WILKENFELD

MAXWELL TECHNOLOGIES, INC  
2 CY ATTN: K ROBERTSON  
2 CY ATTN: M GERSTEN  
2 CY ATTN: MR RAUNCH  
2 CY ATTN: N LOTER  
2 CY ATTN: W RIX

MISSION RESEARCH CORP.  
ATTN: D ALEXANDER  
2 CY ATTN: DAVID SLEETH  
2 CY ATTN: H JAKE TAUSCH

MITRE CORPORATION  
ATTN: J R SPURRIER  
ATTN: M FITZGERALD

ORBITAL SCIENCE CORP.  
ATTN: ROB CHERNEY

PACIFIC-SIERRA RESEARCH CORP.  
ATTN: H BRODE

RAYTHEON CO.  
ATTN: D D LEE  
ATTN: JOSEPH SURRO

RESEARCH TRAINGLE INSTITUTE  
ATTN: M SIMONS

ROCKWELL INTERNATIONAL CORP.  
ATTN: V DE MARTINO

SCIENCE APPLICATIONS INTL CORP  
ATTN: W CHADSEY

SCIENTIFIC RESEARCH ASSOC, INC.  
ATTN: H GRUBIN

SUNDSTRAND CORP.  
ATTN: C WHITE

SYSTRON-DONNER CORP  
ATTN: SECURITY OFFICER

TECHNOLOGY DEVELOPMENT ASSOCIATES  
ATTN: R V BENEDICT

TELEDYNE BROWN ENGINEERING  
ATTN: G R EZELL  
ATTN: LEWIS T SMITH  
ATTN: M P FRENCH

THE AEROSPACE CORP  
ATTN: D SCHMUNK  
ATTN: K G HOLDEN



DSWA-TR-97-15 (DL CONTINUED)

ATTN: LEE MENDOZA  
ATTN: N SRAMEK  
ATTN: R KOGA

THE RAND CORPORATION  
ATTN: C CRAIN

TRW  
ATTN: M J TAYLOR

TRW INC.  
ATTN: T. I. C.

TRW S. I. G.  
STRATEGIC SYSTEMS DIVISION  
ATTN: C BLASNEK

TRW SPACE & DEFENSE SECTOR SPACE &  
TECH GROUP  
ATTN: D M LAYTON

UNISYS CORPORATION-DEFENSE SYSTEMS  
ATTN: P MARROFFINO

VISIDYNE, INC.  
ATTN: C H HUMPHREY  
ATTN: W P REIDY